

CEMAP: General Usage Guide

Terrill L. Frantz & Kathleen M. Carley

May 15, 2009
CMU-ISR-09-116

Institute for Software Research
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213



Center for the Computational Analysis of Social and Organizational Systems
CASOS technical report.

This work is part of the Dynamics Networks project at the center for Computational Analysis of Social and Organizational Systems (CASOS) of the School of Computer Science (SCS) at Carnegie Mellon University (CMU). This work is supported in part by the Office of Naval Research (ONR MMV - N00014-06-1-0104), United States Navy. Additional support was provided by National Science Foundation (NSF) Integrative Graduate Education and Research Traineeship (IGERT) program, NSF 045 2598, NSF 045 2487, and CASOS. MURI(MURI (AFRSO)GMU – FA9550-05-1-0388): Air Force Office of Scientific Research (ARL Telcordia – 20002504; ARL Alion – 1193343TO102), FA9550-05-1-0388 for Computational Modeling of Cultural Dimensions in Adversary Organizations. SPAWAR, Network Analysis and Computational Modeling for Combating Terrorist Threats, MOAT Phase II, DNA application to counter-narcotic investigations related to marijuana. The views and proposal contained in this document are those of the author and should not be interpreted as representing the official policies, either expressed or implied, of the Office of Naval Research, the National Science Foundation, or the U.S. government.

Report Documentation Page				Form Approved OMB No. 0704-0188	
Public reporting burden for the collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.					
1. REPORT DATE 15 MAY 2009		2. REPORT TYPE		3. DATES COVERED 00-00-2009 to 00-00-2009	
4. TITLE AND SUBTITLE CEMAP: General Usage Guide				5a. CONTRACT NUMBER	
				5b. GRANT NUMBER	
				5c. PROGRAM ELEMENT NUMBER	
6. AUTHOR(S)				5d. PROJECT NUMBER	
				5e. TASK NUMBER	
				5f. WORK UNIT NUMBER	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Carnegie Mellon University,School of Computer Science,Institute for Software Research,Pittsburgh,PA,15213				8. PERFORMING ORGANIZATION REPORT NUMBER	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)				10. SPONSOR/MONITOR'S ACRONYM(S)	
				11. SPONSOR/MONITOR'S REPORT NUMBER(S)	
12. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution unlimited					
13. SUPPLEMENTARY NOTES					
14. ABSTRACT see report					
15. SUBJECT TERMS					
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT Same as Report (SAR)	18. NUMBER OF PAGES 74	19a. NAME OF RESPONSIBLE PERSON
a. REPORT unclassified	b. ABSTRACT unclassified	c. THIS PAGE unclassified			

Keywords: CEMAP, source data, social network, ORA, AutoMap, software, DyNetML, dynamic network analysis

Abstract

CEMAP software is a wide-ranging solution to the many practical and logistical problems that analysts face when performing the basic data processing steps necessary for using analytic software. CEMAP transforms real-world source data into a form that is specifically formatted to the exacting requirements of particular social network analysis software. This report provides the concepts important to CEMAP as well as step-by-step instructions on how an analyst uses CEMAP as either a stand-alone or ORA/Automap-hosted interactive system, or as an automated command-line batch system. Taking the short time to learn how to use CEMAP, which is easy, will save a great deal of time in one's workflow within the analytic process and will put a wealth of popular real-world datasets within easy reach. This report guides the reader through the major aspects of CEMAP from the perspective of the role the operator plays in the work flow from raw-data extract, transformation, and the loading of network data, for subsequent analysis.

Summary Table of Contents

1	Introduction	1
2	A Quick Start	8
3	Getting Started as a User: Managing Profiles	50
4	Getting Started as an Analyst: Managing Templates	56
5	Getting Started as a Developer: Managing Tablesets.....	61
6	Forward.....	65
7	References	66

Detailed Table of Contents

1	Introduction	1
1.1	System Overview	1
1.2	Usage Roles.....	2
1.2.1	User	3
1.2.2	Analyst	3
1.2.3	Developer	3
1.3	Starting CEMAP	3
1.3.1	Within ORA or AutoMap	4
1.3.2	Stand-alone Execution	5
1.3.3	Command-line Execution	5
1.4	Execution modes	6
1.4.1	Interactive	6
1.4.2	Non-interactive	6
2	A Quick Start	8
2.1	Loading and Executing a Profile.....	9
2.2	Modifying a Profile	17
3	Getting Started as a User: Managing Profiles	50
3.1	Description of a Profile	50
3.2	Managing Profiles	50
3.2.1	Loading a profile.....	50
3.2.2	Saving a profile	51
3.2.3	Modifying a profile	52
3.3	Executing a Profile	53
3.3.1	Input Fields	53
3.4	The Start-up Profile.....	55
3.5	Mapping a Template to a Tableset	55
3.6	Executing a Profile directly from the Command-Line.....	56
4	Getting Started as an Analyst: Managing Templates	56
4.1	Description of a Template	56
4.2	Managing Templates	57
4.2.1	Template Fields.....	59

4.3	Template Types	60
4.3.1	DyNetML	60
4.3.2	Directory	60
4.3.3	Directory DyNetML	61
5	Getting Started as a Developer: Managing Tablesets	61
5.1	Description of a Tableset	61
5.2	Managing Tablesets	62
5.2.1	Tableset Fields	63
5.3	Using Resources	65
6	Forward	65
7	References	66

1 Introduction

In software engineering parlance, CEMAP (see Frantz & Carley, 2008a) is software that belongs within the family of Extract, Transform, and Load (ETL) systems. Its purpose and its architecture are designed to extract data from an electronic source in a particular data format then transform the data into a format that makes it possible to load into a target software system. The chief rationale for providing CEMAP is the recognition that the ETL process is not a value-add to the network analysis process, though it is a vital activity that must be carried out: the process of moving data between software systems is usually time-consuming and sometimes quite frustrating. Moreover, the technological skills necessary to perform this meticulous task are sometimes personally out of reach to develop, or costly to farm out to a skilled programmer or purchase. CEMAP helps the network analyst to overcome these problems and situates them to allocate greater personal time and energy to the more important activity of performing data analysis, rather than spending valuable time on data manipulation tasks.

CEMAP is a tool that is created especially to assist users of ORA (Carley & Reminga, 2004) and AutoMap (Diesner & Carley, 2004), but it can be used to aid users of other similar software. CEMAP can be used in either an interactive or a hands-off batch mode. In the interactive mode, users can design the output datasets in a repetitive fashion with minimal effort—the software is designed with the realities of the analytic workflow in mind. As a batch process, CEMAP can be set up to execute in an automated fashion, for example, to extract one’s email network every night at midnight.

1.1 System Overview

CEMAP cannot eliminate the unavoidable requirement for executing the technical detailed steps necessary to extract, transform and load data. Instead, CEMAP re-organizes the ETL process into simpler sub-component ETL steps. The individual steps segregate much of the *technical* detail of the source data from the *analytic* details of the network data to be loaded. This separation reduces the expanse of the requirements for skills necessary for a individual to use CEMAP; individuals use only the parts of CEMAP that they understand, but can still benefit from the full functionality of CEMAP. As Fig. 1 depicts, the master ETL process performed by CEMAP is partitioned into two separate, but connected ETL processes. In effect, the two sub-processes serve to separate the data-technical aspects of the higher-level task from the network-analytic aspects of the higher-level, master ETL process.

The inner workings of the two sub-systems and their intrinsic user interface involve three vital elements: templates, tablesets and the template/tableset mapping. When brought together and correctly aligned, these three elements form what is known as a *profile*. A CEMAP profile is an all-encompassing component that provides CEMAP with what it needs to execute the high-level ETL process. A *template* describes the end-process, output files that CEMAP is being asked to create for later loading into the downstream or host software. A *tableset* describes the technical elements necessary to extract the raw source data. This tableset-to-template mapping contained within a profile expresses the manner in which the output of the tableset is logically connected to the input of the template. Critically, the output of the tableset is a set of one or more relational data tables (think excel spreadsheet) and the input to the template is a set of one

or more relational data tables. The mapping simply describes the connection between the tableset output and the template input. The profile contains all of this information in one logical construct.

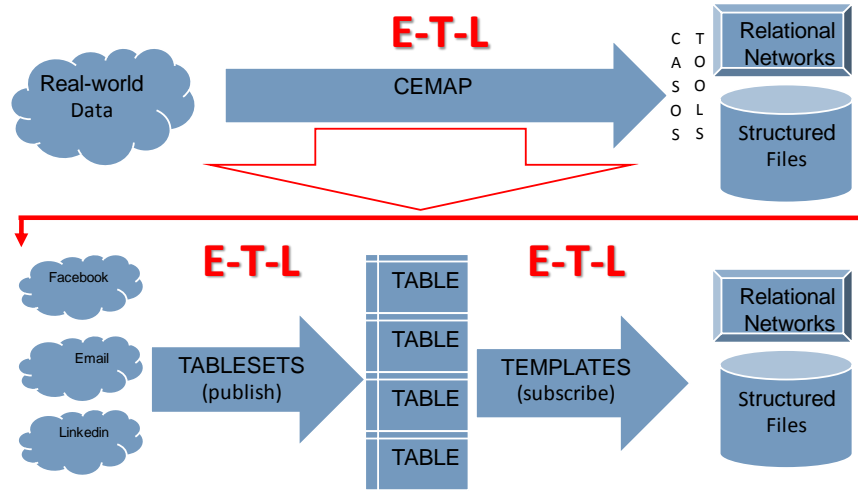


Figure 1. CEMAP is partitioned into two separate and distinct ETL processes so that network analysts need not also be programmers, and vice versa.

This architecture situates the CEMAP system into becoming a system that has a user-experience that is motivated and driven by a task-view. This is to say that when one uses CEMAP, they should think about what the sub-task is that they need to perform and need only to have the perspective of that particular task to perform it. For example, when designing the output that needs to be constructed for the downstream or client software system, using CEMAP one needs to focus only on the output design and pay very little attention to the format of the original source data. Conversely, when one is attending to the technical details of extracting raw, source data, they need not be concerned with the specifics of the down-stream, output format. Skill-wise, tablesets are the domain of technology-oriented programmers, and templates are the domain of the analytic-oriented, network analysts. Profiles are the concern of a person in the role of an end-user, who may not have any interest in the details of either the source data, or the data being loaded, but is interested in using the data in the downstream software. This is to say that when developing a Tableset, programmer skills are necessary; when developing a template, analyst skills are necessary; and, when executing a profile, only basic skills (from the perspective of CEMAP) are called for. These usage roles are further explained in the following sub-section.

1.2 Usage Roles

A distinguishing feature of CEMAP is that rather than being designed only from a technological standpoint--as most software is architected--it is also designed from a *social* standpoint. The usage of CEMAP is based on three explicit usage roles: *User*, *Analyst*, and *Developer*. Each role reflects the manner in which a person would typically be using CEMAP. There are situations where a single person uses CEMAP in only one of these roles, or perhaps in more than one of these roles. The roles are designed to match the purpose that a person would be using CEMAP at a particular time, as well as to

accommodate the background and skill set of the person using the tool. Respectively, from User, Analyst, to Developer, the role becomes more technical and detailed in nature and skill expectations. These three roles are explained further in the subsections that follow.

1.2.1 User

An operator using CEMAP in the role of a *user*, generally, has all the necessary tableset and template configuration and mapping information already set up for them in the form of a Profile. The user typically needs to merely identify and select a Profile, then execute it. Such simplicity embedded in the profile, makes the actual usage of CEMAP somewhat of a trivial task, and as such, a new user requires minimal training on how to use CEMAP. The user can instead focus on performing analysis on the data rather than concerning themselves with the operational details of obtaining the data. Simply they can “point and execute.”

Depending on the specific profile the user is executing, the operator may be presented with some windows that optionally request, or require, the operator to enter in some relevant data before continuing with the execution of the profile. Such information can be routine--such as indicating where an output file should be stored on their disk, if desired or perhaps a user password necessary to access the data they are looking to access. CEMAP has the capability to store this entry information, including passwords, with a custom profile. Therefore a fully-completed profile, with all its execution information stored within, can be executed by an operator without having to redundantly enter in any information; it is necessary, then, to complete the entire ETL process with just a few clicks of the mouse.

1.2.2 Analyst

An operator using CEMAP in the role of an *analyst*, generally, is concerned with (a) setting up and maintaining CEMAP Templates, and (b) performing the mapping process necessary to direct Tableset data to the Templates describing the desired output files. This role necessitates the person to thoroughly understand the network/relational structure of the out data that the host software expects as input and any other files that are to be produced as an outcome of running the CEMAP tool. Little knowledge of the underlying technicalities of the tablesets is necessary. The developer (see below) is charged with constructing the tableset with the technicalities hidden and with sufficient operator notes—CEMAP facilities both necessities.

1.2.3 Developer

An operator using CEMAP in the role of a *developer*, generally, is concerned with setting up and maintaining CEMAP Tablesets. This role necessitates the person to thoroughly understand the technicalities of accessing the source data from a programming perspective. This role does not require any knowledge of network analysis of the subtleties of network data files.

1.3 Starting CEMAP

CEMAP can be started either as a subsystem within a host software program, specifically ORA or AutoMap, or as an independent, stand-alone program.

1.3.1 Within ORA or AutoMap

To run as a hosted subsystem, it is started via a menu item in the host software menu. Figure 2 shows the start-up menu selection for ORA. This is the File>CEMAP menu path. Figure 3 shows the working-screen for CEMAP in the interactive mode that comes about when the CEMAP menu item is selected.

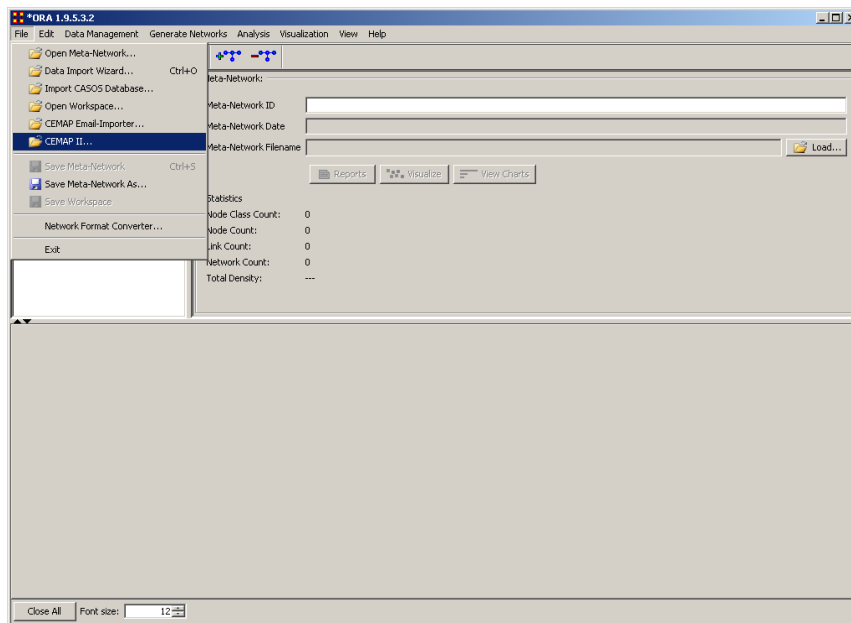


Figure 2. Starting CEMAP from the ORA host software.

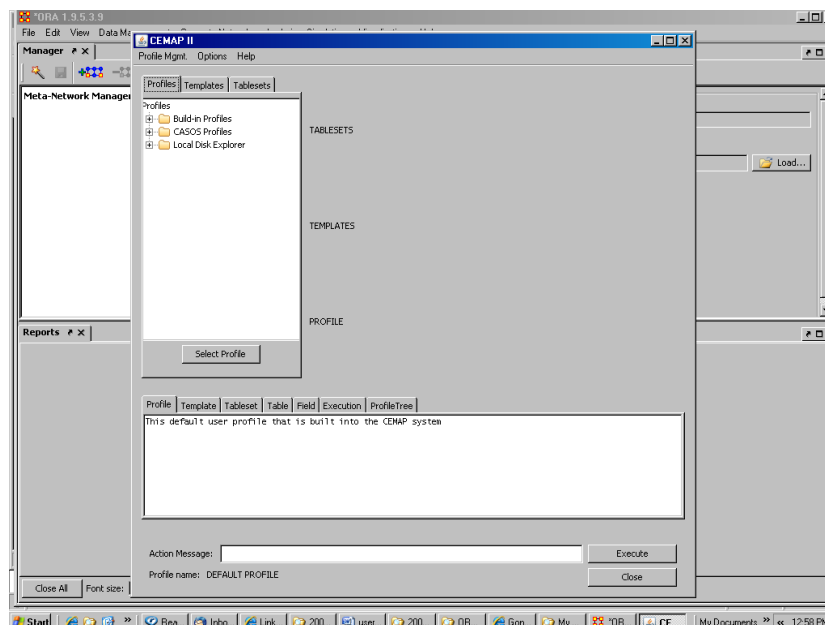


Figure 3. The CEMAP workspace when started from ORA in interactive mode.

1.3.2 Stand-alone Execution

CEMAP can also be executed independent of a host software system. CEMAP software is a windows menu item (see Figure 4). The CEMAP GUI will be displayed and the program is executed identically to the hosted method. The is one functional differences, however: in the ORA hosted version, any DyNetML (Tsvetovat, Reminga & Carley, 2003). files created in CEMAP will be loaded into ORA—the standalone version will not implement this convenience.

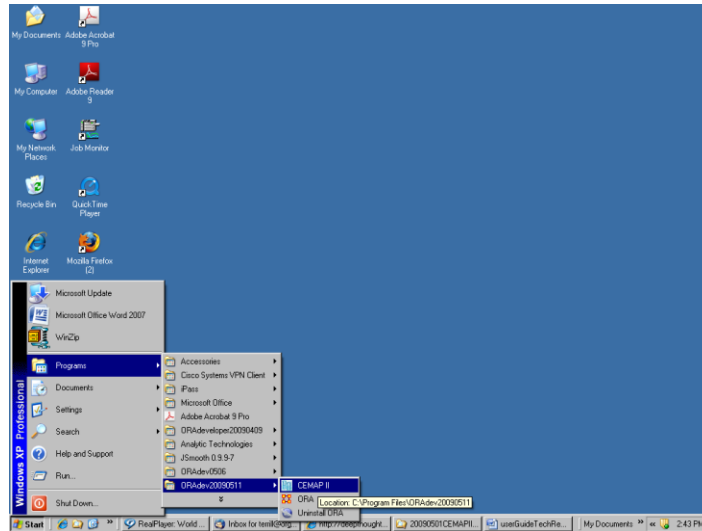


Figure 4. Start running CEMAP from a Windows programs menu.

In addition to the ORA Windows menu item for running CEMAP interactively, the ORA Windows menu also has an item to place the operator into the CEMAP software root directory for easy execution on the MS-DOS command line. While getting to the command line in this manner is convenient, it is suggested that frequent CEMAP command-line operators put the ORA root directory in the Windows (or other operating system) command-line PATH/path environment variable.

1.3.3 Command-line Execution

CEMAP can also be executed independent of a host software system. CEMAP software is automatically installed with ORA software so the program start-point is the cemap.exe program file located in the home file folder of the installed ORA software, which is typically the folder C:\Program Files\ORA. To execute CEMAP as a stand-alone system, simply run cemap.exe by double clicking or running the executable file from a MS-DOS window by typing either cemap.exe or cemap; this will display the same CEMAP workspace as shown in Fig. 3 and be available for interactive use. CEMAP run as a standalone is identical to running as a sub-process within a host; however, as a sub-process, it is equipped to automatically load any network files into the ORA meta-network workspace as well as any outputs specified in the profile; all other aspects of the CEMAP workspace are identical.

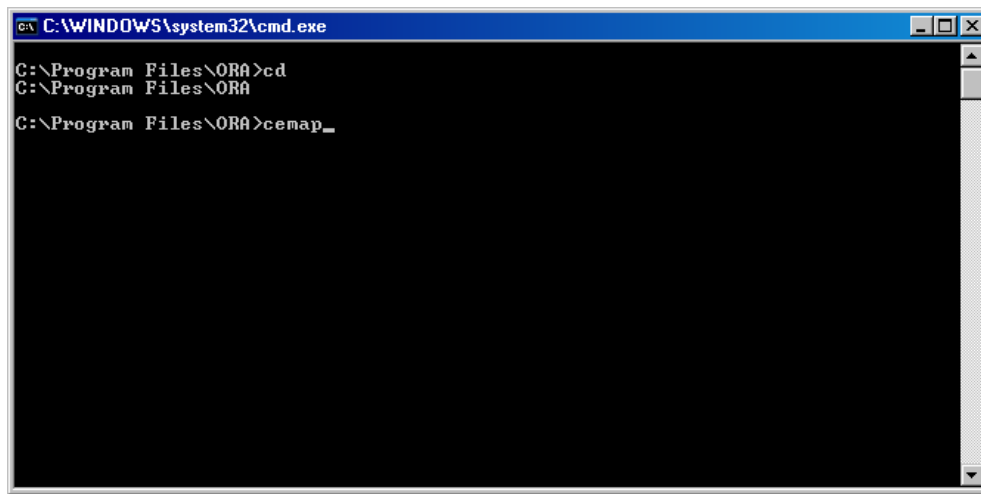


Figure 5. Execute CEMAP from MS-DOS command line.

1.4 Execution modes

CEMAP executes either one of two modes: interactive or batch. Multiple instantiations of CEMAP can be run simultaneously, though the operator should be careful about output filename collisions.

1.4.1 Interactive

The interactive, Graphical User Interface (GUI), instantiation of CEMAP provides the typical, non-programmer, with the full-feature set of CEMAP functionality, but with a friendly, easy to use operator interface. This interface involves interactive panels within which the operator maintains profiles, templates, and tablesets. Moreover, the interactive interface simplifies the mapping of templates to tablesets, maintaining operator notes, and executing profiles.

At the technology-level, the GUI simply makes it easy for the operator to edit the profile and execute the profile interactively. The underlying purpose of the GUI is to create an XML file that is actually the CEMAP profile. The operator can, just as effectively, modify a profile by using an XML editor; though, using the GUI reduces the need to understand the details of the profile xml document. The subtleties of the profile xml document are masked by the GUI interface.

To execute CEMAP in interactive mode, the operator can either start the program via an operating system menu, or via command-line execution; these capabilities are described in the subsections above.

1.4.2 Non-interactive

CEMAP can be executed as a batch process from the operating system command line. From the command line execution, it can execute the usual GUI workspace, or can immediately execute a specified profile. From the command line, if `cemap.exe` (or alternatively `CEMAP`) is entered without any command line parameters, the GUI mode

will be activated and the usual CEMAP workspace window will appear. Alternatively a profile file can be stdin piped in, or be added to the command line, which will cause that profile to be executed immediately. Following the completion of the process indicated by the profile, CEMAP execution will end. Using the `-h` or `--help` switches on the command line will display the currently available switches and command line format. The stdin pipe capability is provided to aid programmers in using text-altering programs such as `sed` to modify the profile being supplied, e.g. change the output filename, in a programmatic fashion. This is particularly useful if CEMAP is to be run in an automated and routinely scheduled (hourly, daily, weekly, etc.) manner as a cron job, etc.

The syntax for command line execution of CEMAP is:

```
cemap[.exe] [parameters] [profileFiles] < [profileXMLStream]
```

The syntax for command line execution *using pre-cmd stdin piping* is:

```
type [profileXMLStream] | cemap[.exe] [parameters] [profileFiles]
```

```
cat [profileXMLStream] | cemap[.exe] [parameters] [profileFiles]
```

Parameters include:

- `-h, --help` displays the current parameters and command line syntax
- `-p, --profile` loads the profile into memory at the start of cemap gui
- `-i, --interactive` executes CEMAP in the interactive gui mode
- `-g, --gui` (if profile is supplied, the first listed (or stdin) will be loaded into memory at the start)
- `-a, --automatic` executes CEMAP in the cycle-through mode according to the Profile Files, or profile stdin supplied.

`profileFiles` is a file that consists of a completed profile. This file can be either a local file or a file available via a URL (http or https). This will load and execute the profile, then end CEMAP. (multiple profiles can be provided—this will result in multiple independent load/execute cycles of CEMAP.

`profileXMLStream` is a stdin pipe that consist of characters that form a complete profile file. This will load and execute the profile, then end CEMAP. (this suggests the `--auto`

If cemap is called alone, without any command-line parameters, a profileFile, or a stdin pipe, the (`--help` option) will be displayed.

2 A Quick Start

We begin this quick start into CEMAP by walking through a simplistic introduction to the process that a user would go through to use CEMAP. First we show how to load one of the CASOS public datasets into ORA, then we walk through the process of loading email into ORA. Most likely the ORA-hosted process will be a person's first exposure to CEMAP, so it is a natural starting point for most people. We provide these two tasks that are realistic and simplistic in their source because we want to focus on the CEMAP aspects of the task rather than non-CEMAP issues. After working through these two examples, we suggest that you look at the CEMAP Facebook technical report (Frantz & Carley, 2008c) for a more exciting walk-through; a personal Facebook account will be necessary for you to run the Facebook tutorial.

There are two basic, sequential steps to processing data using CEMAP:

- I. Load a Profile
- II. Execute the Profile

The first step, Load a Profile, means that a profile is to be loaded into CEMAP's working memory. Once a profile is loaded into CEMAP memory the contents of the profile can be explored, changed, saved, replaced, printed, or executed. The second step initiates the actual execution of the profile, putting the process into action. For both examples in this section we are assuming that the operator is already running and in ORA and has started the CEMAP GUI, and therefore see the window as shown by Figure 6; this is the starting-point for both examples.

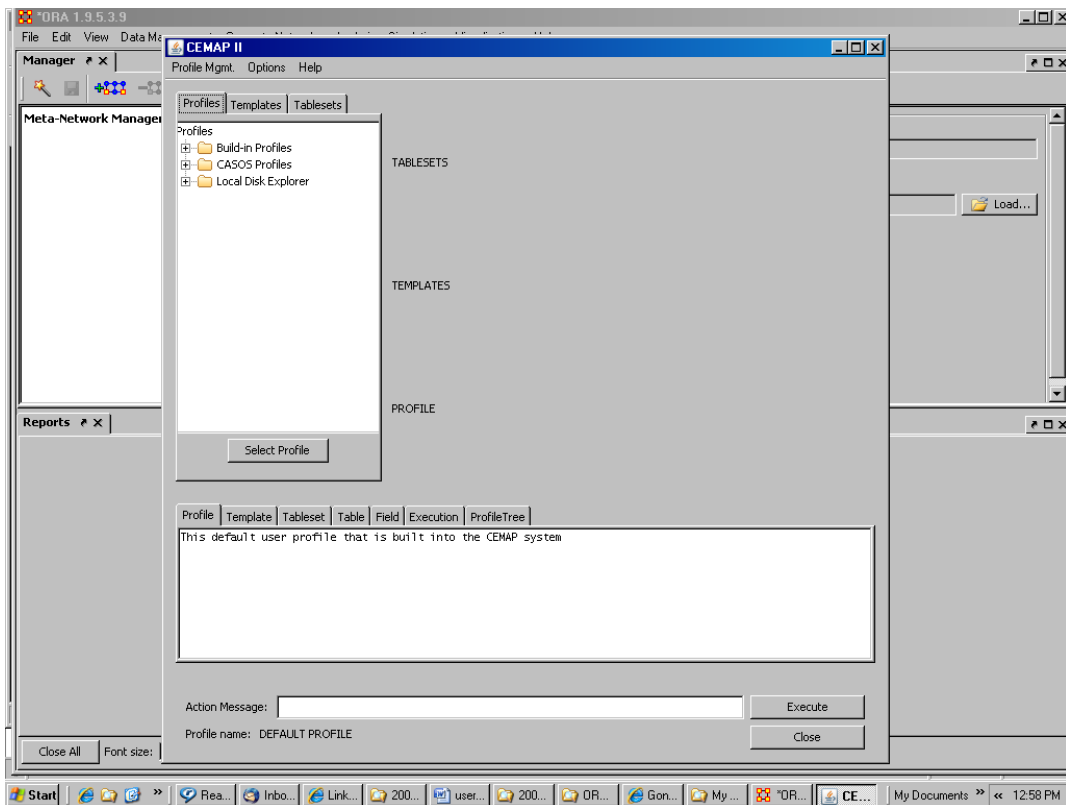


Figure 6. The CEMAP workspace when started from ORA in interactive mode.

2.1 Loading and Executing a Profile

CASOS has publicly available datasets that have been used in past scholarly research. These datasets are made available via the CASOS web site, Tools, Models, and Data page, which is viewable at this URL:

http://www.casos.cs.cmu.edu/computational_tools/data2.php

We use for this example the Zachary data set, which is publicly available at this page:

http://www.casos.cs.cmu.edu/computational_tools/datasets/external/karate

Without CEMAP, the general process one will follow is to: (a) go to the above URL in an Internet browser, (b) locate a place on the local computer to save the data (c) extract the dataset to that location on the local computer disk, and then (d) return to ORA and (e) perform a file-open process (which includes re-locating the local data file). Using CEMAP, the general process is to (a) locate and load the Zachary dataset profile, and (b) press the *Execute* button.

To load the Zachary dataset into ORA using CEMAP follow these steps:

Step 0:

Task: Start the CEMAP GUI

Action: Start CEMAP GUI in ORA

Result: you will see the CEMAP GUI workspace, e.g. Fig. 7.

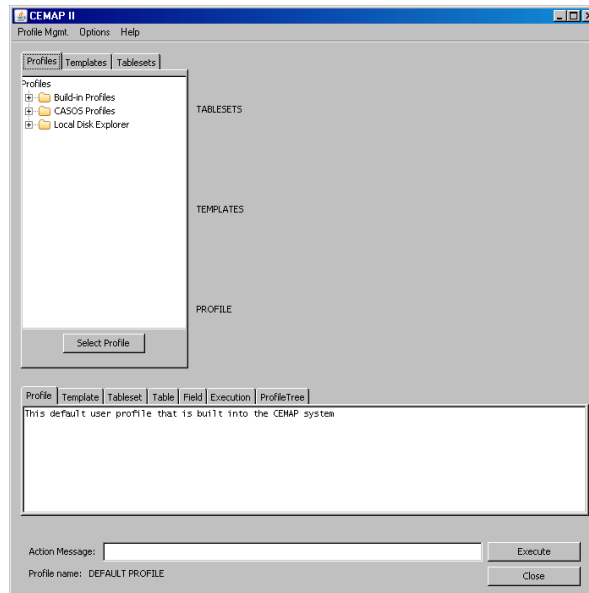


Figure 7. Result of Step 0.

Step 1:

Task: Locate the profile for the Zachary dataset

Action: Locate the Zachary profile in the Profiles workspace (under Built-in-Profiles/General) and highlight the Zachary entry.

Result: you will see the screen

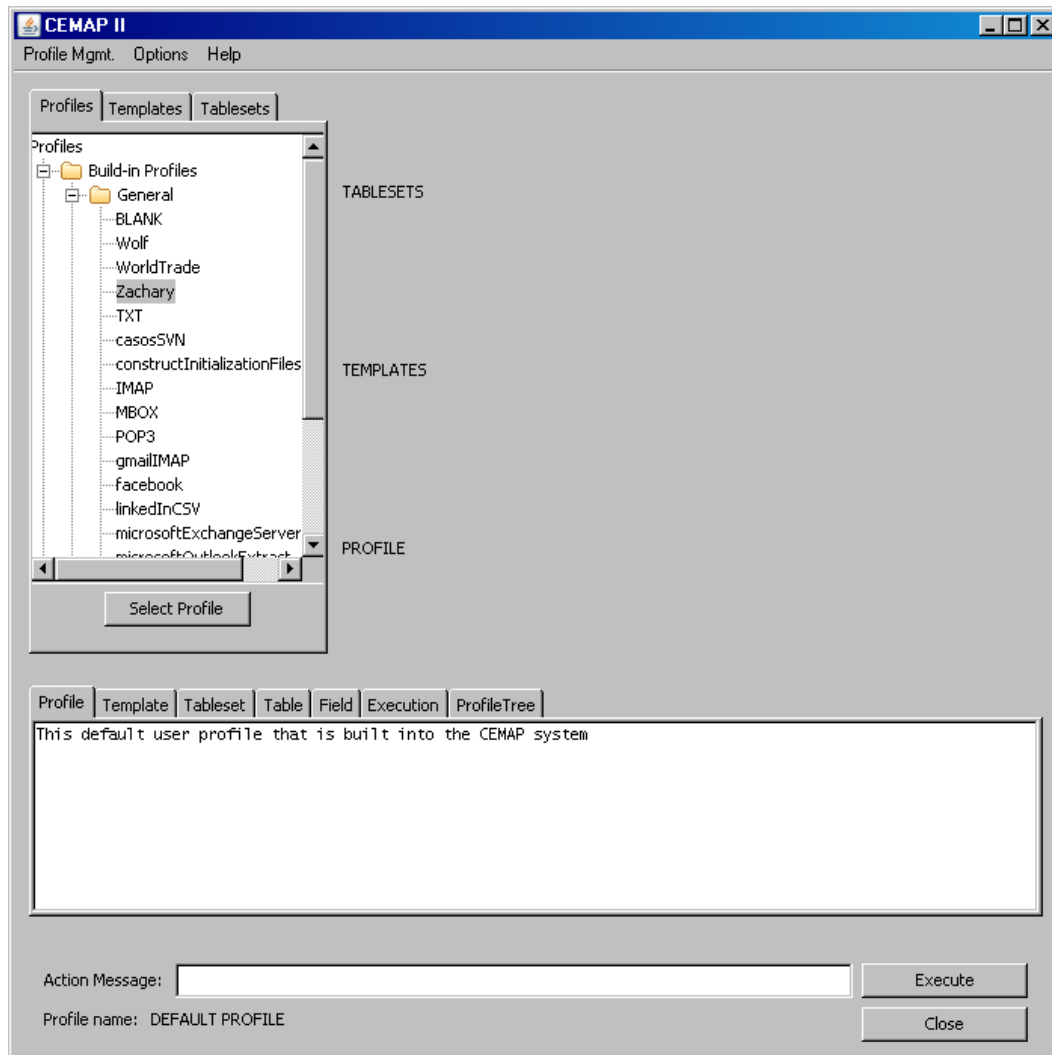


Figure 8. Result of Step 1.

Step 2:

Task: Load the profile into CEMAP memory

Action: Load the Zachary profile by clicking the Select Profile button below the Profiles Window .

Result: You will see the text in the Profile Message window change to match the Zachary profile – these are notes about the dataset that users may be interested in.

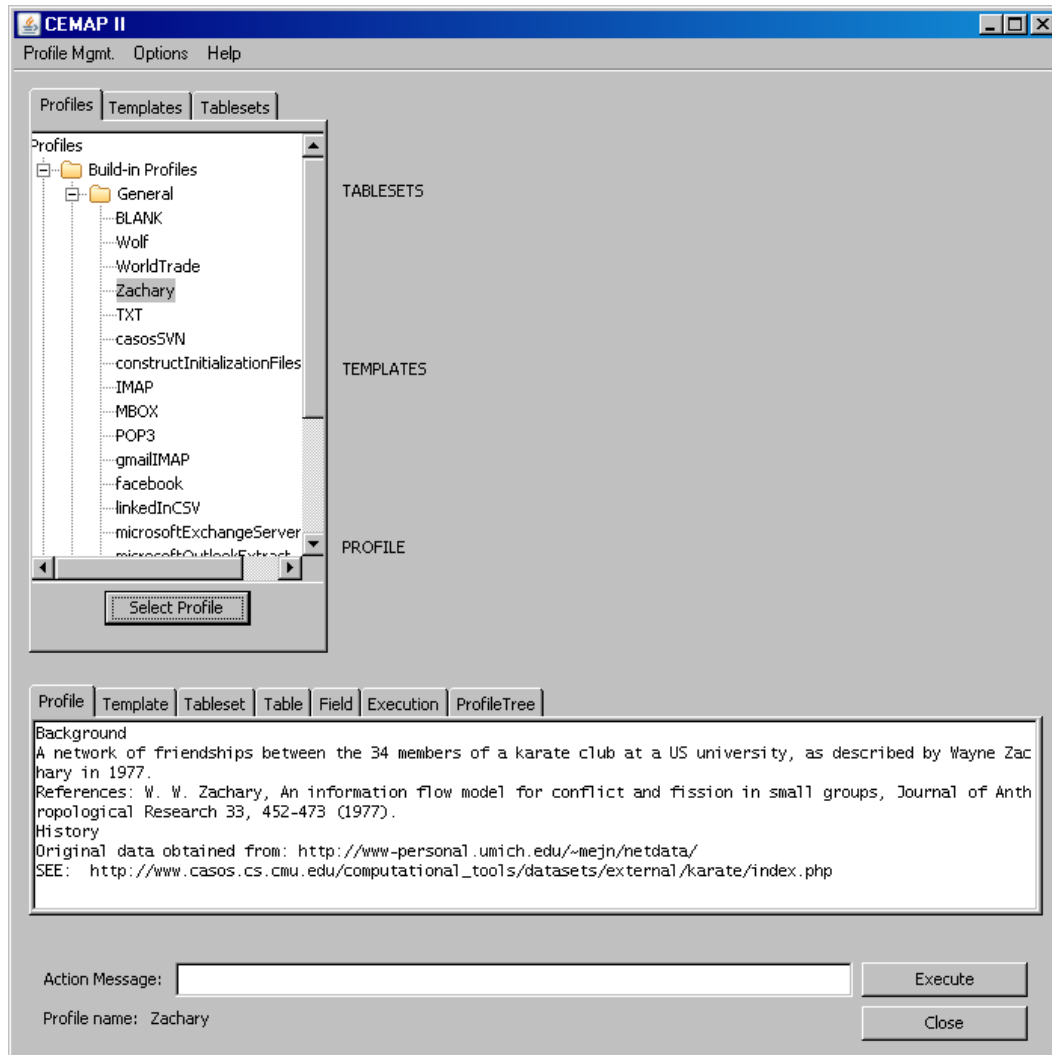


Figure 9. Result of Step 2.

Step 3:

Task: Begin the execution of the profile

Action: Click the Execute button on the lower-right portion of the CEMAP window.

Result: You will see (a) the Execute Message Window over-display the Profile Message Window, thus you will see a message with a yellow background, and (b) a smaller-sized dialog window displayed as shown in Figure 10.

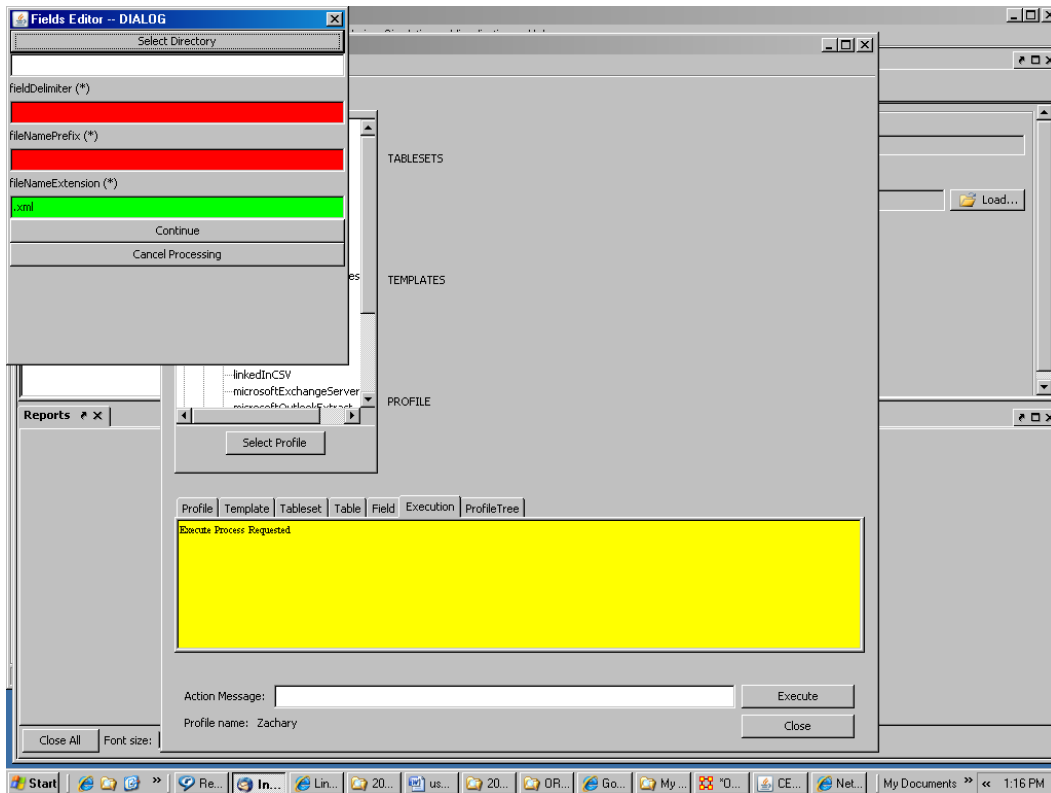


Figure 10. Result of Step 3.

Step 4:

Task: Respond to data fields that require, or expect, input. For the purposes of this walk-through, we will not save the dataset to a local disk file; this is why we simply continue at this step, rather than identifying a location where the file should be also saved.

Action: Press the “Continue” button.

Result: You will see some screen flashes (this is fast so you will miss it, but is shown in Figure 11) followed by a message saying Parsing Complete meaning that the execute process is completed--as shown in Figure 12.

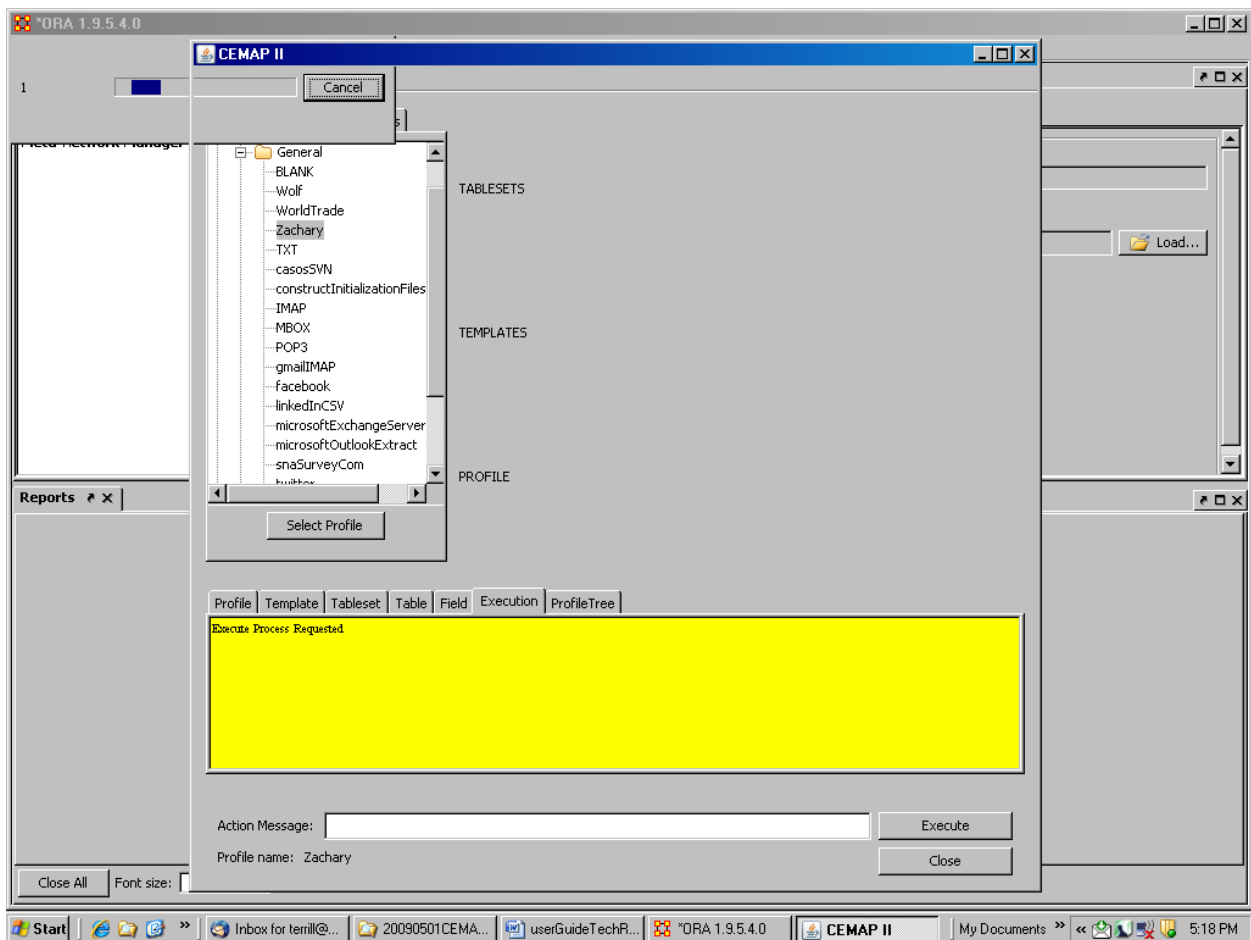


Figure 11. Mid-step process during Step 4 – progress bar shows every briefly on the screen.

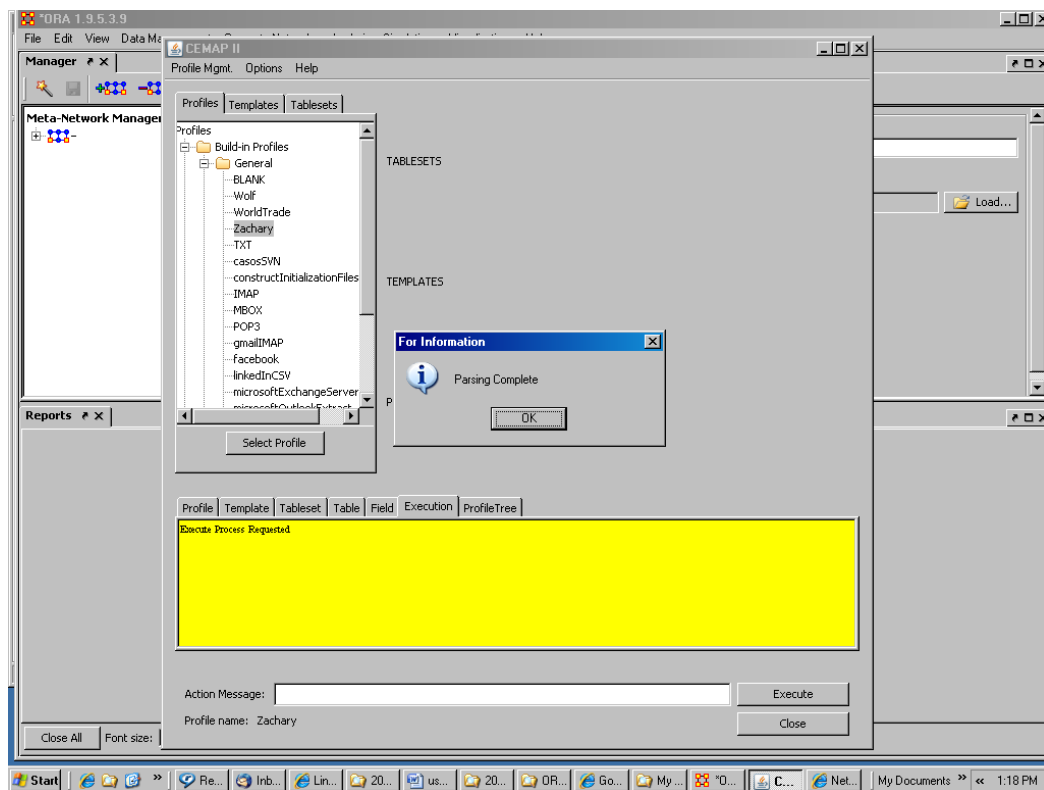


Figure 12. Result of Step 4.

Step 5:

Task: Acknowledge the completion of the execute request

Action: press the parsing complete Ok button

Result: Now, look at the ORA meta-network window and you will find the Zachary dataset already loaded and immediately available to ORA. The CEMAP workflow is complete

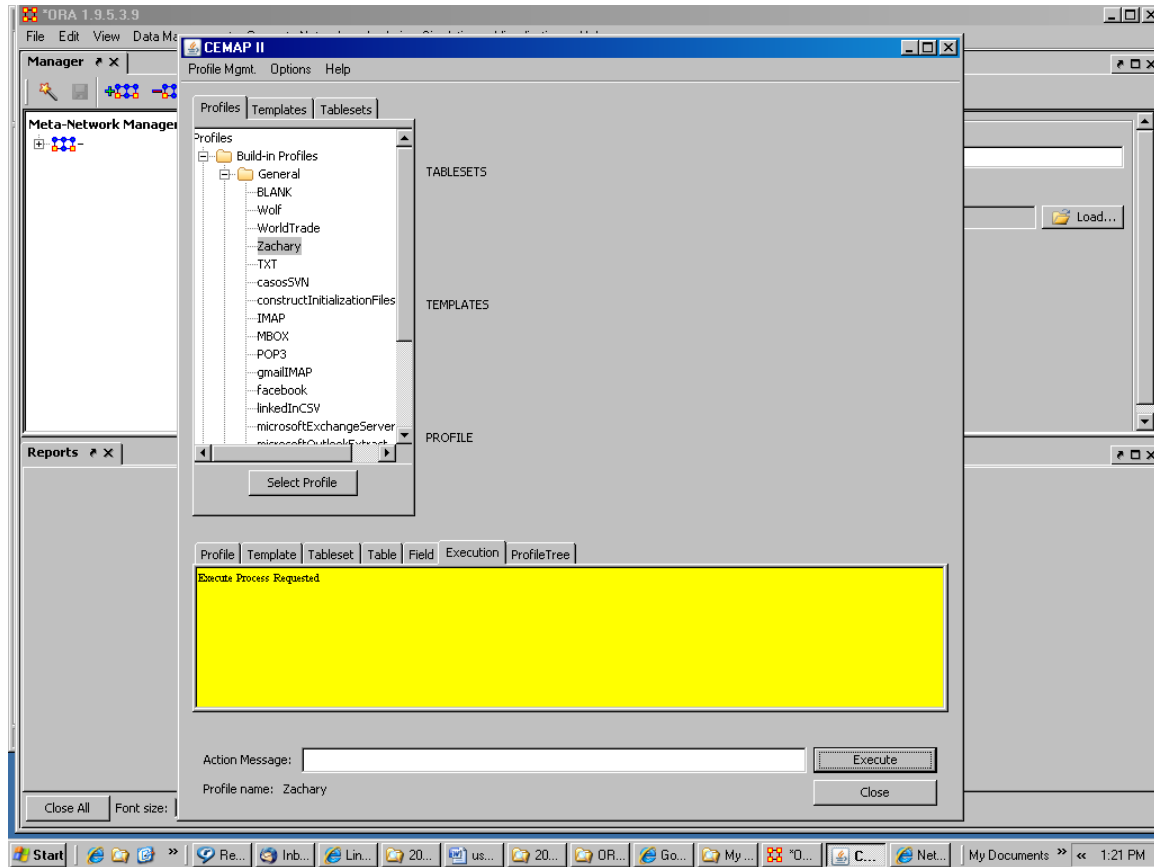


Figure 12. Result of Step 5.

Step 6:

Task: End the CEMAP GUI session

Action: press the Close button.

Result: CEMAP session has ended and the GUI window is closed, you can do as you wish with the data in ORA. Keep in mind that these steps that we took in this example do not save the dataset on your local disk; the data currently resides only in ORA memory. You may want to save the meta-network via the ORA meta-network saving functionality. (note that ORA does not display a meta-network name for this dataset – there is no name in the original dataset and CEMAP will not add one unless specifically instructed to do so.)

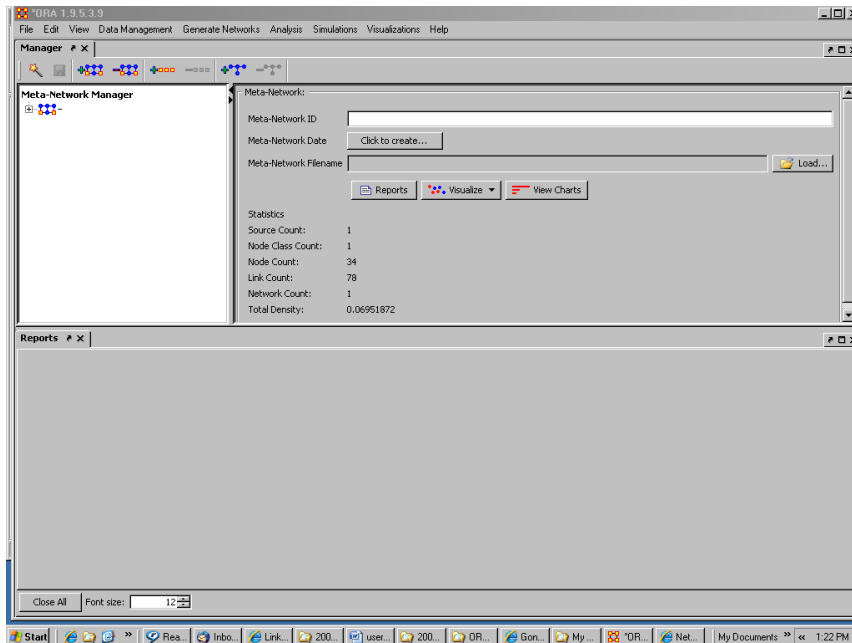


Figure 13. Result of Step 6.

2.2 Modifying a Profile

In this example we walk the operator through the process of loading a personal email account into ORA, via CEMAP. This example is specific to the GMAIL email hosting system, but other email servers can be accessed with minimal differences. The purpose of this example is to illustrate the process of modifying and saving a profile, rather than how to load and execute a profile (as illustrated in the prior walk-through).

To load, change, and execute a profile using CEMAP follow these steps:

Step 0:

Task: Start the CEMAP GUI

Action: Start CEMAP GUI in ORA

Result: you will see the CEMAP GUI workspace, e.g. Fig. 14.

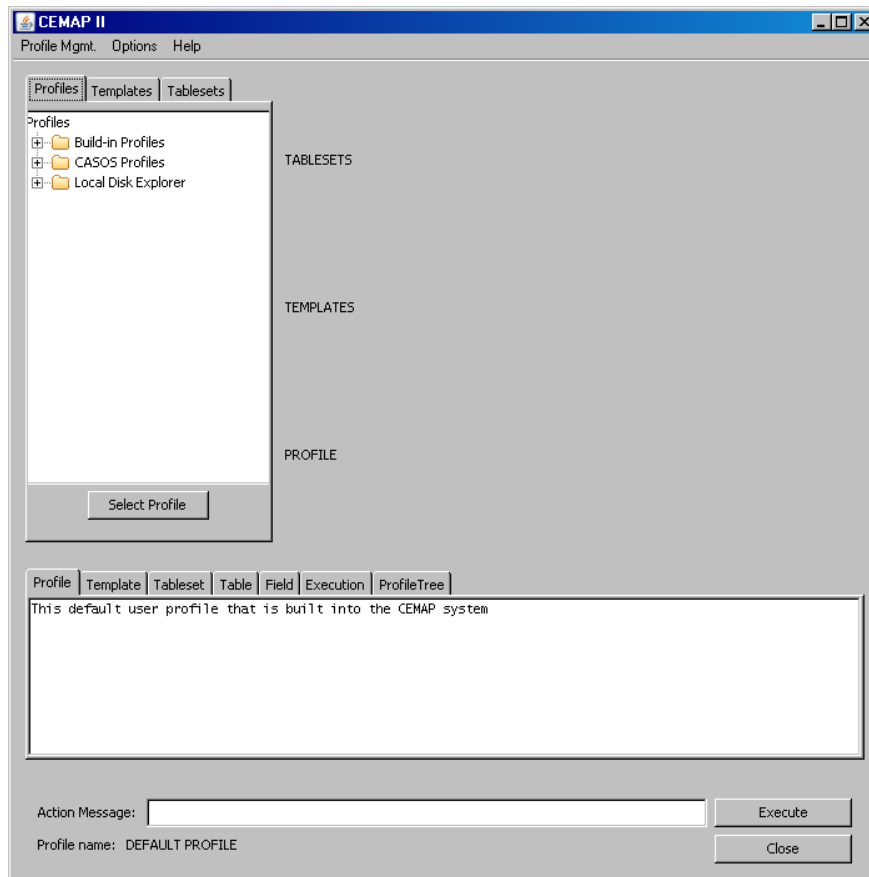


Figure 14. Result of Step 0.

Step 1:

Task: Locate the GMAIL profile, part I

Action: Find the Built-in profiles item and expand it using the left click on the mouse when pointing at the + symbol

Result: The item tree will expand and you will see the CEMAP GUI workspace, e.g. Fig. 15.

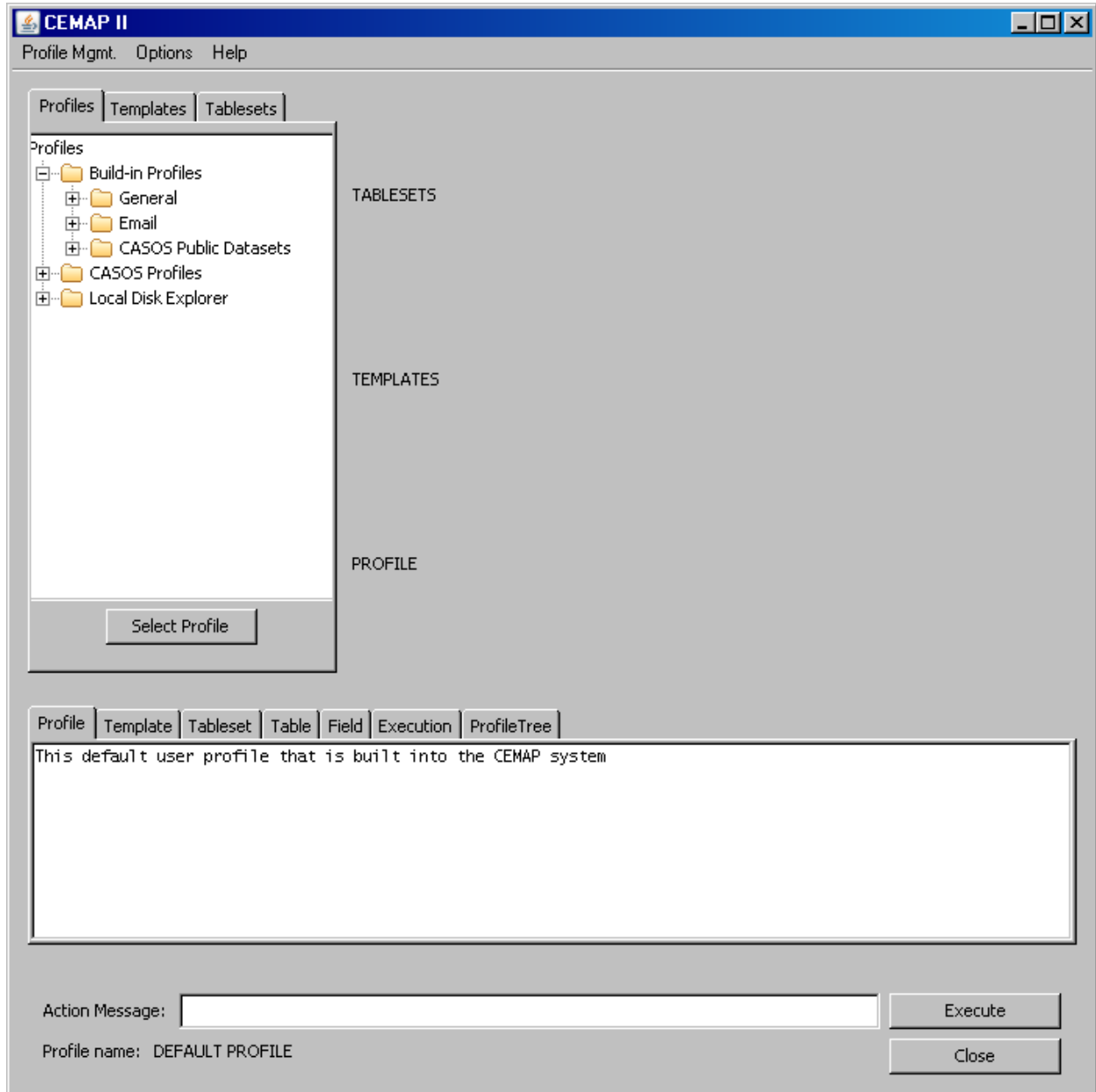


Figure 15. Result of Step 1.

Step 2:

Task: Locate the GMAIL profile, part II

Action: Find the Email profiles item under the Built-in Profiles and expand it using the left click on the mouse when pointing at the + symbol

Result: The item tree will further expand and you will see the CEMAP GUI workspace, e.g. Fig. 16.

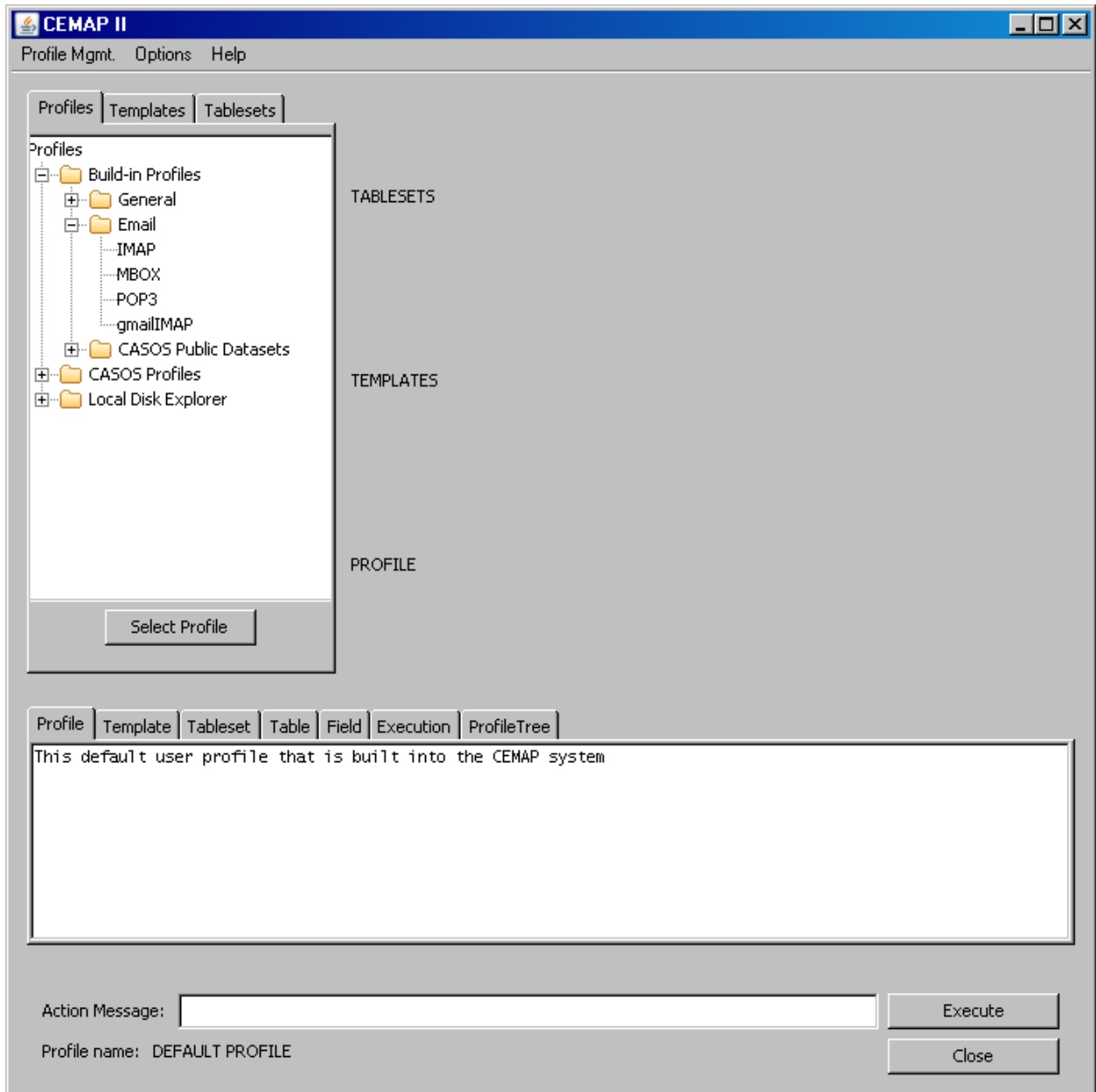


Figure 16. Result of Step 2.

Step 3:

Task: Locate the GMAIL profile, part III

Action: Left click on the mouse when pointing at the gmailIMAP item

Result: The gmailIMAP item will highlight, e.g. Fig. 17.

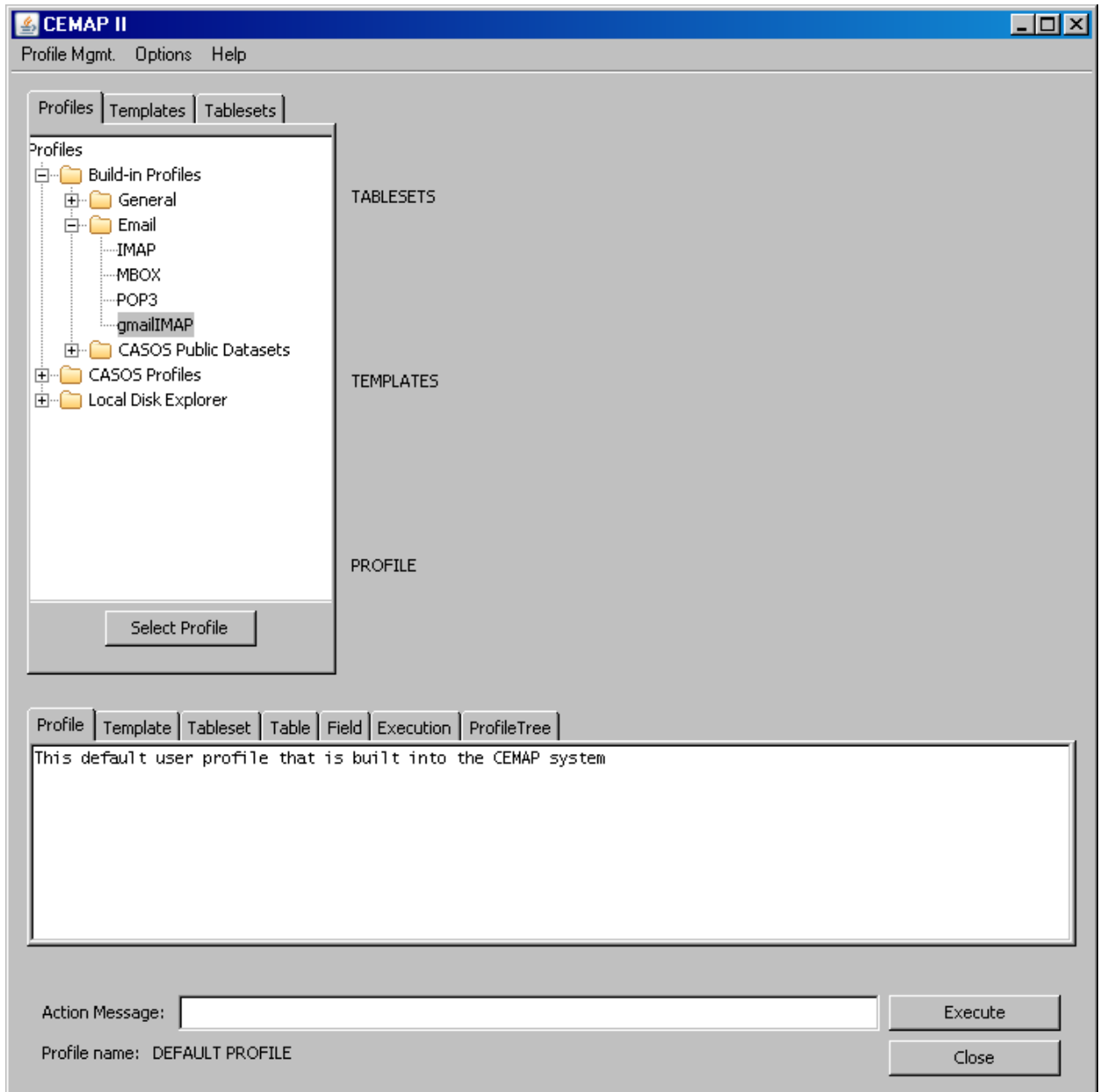


Figure 17. Result of Step 3.

Step 4:

Task: Load the GMAIL profile into active memory

Action: Left click Select Profile button

Result: The text in the Profile message window on the bottom will change, e.g.

Fig. 18. The profile is now in CEMAP's working memory.

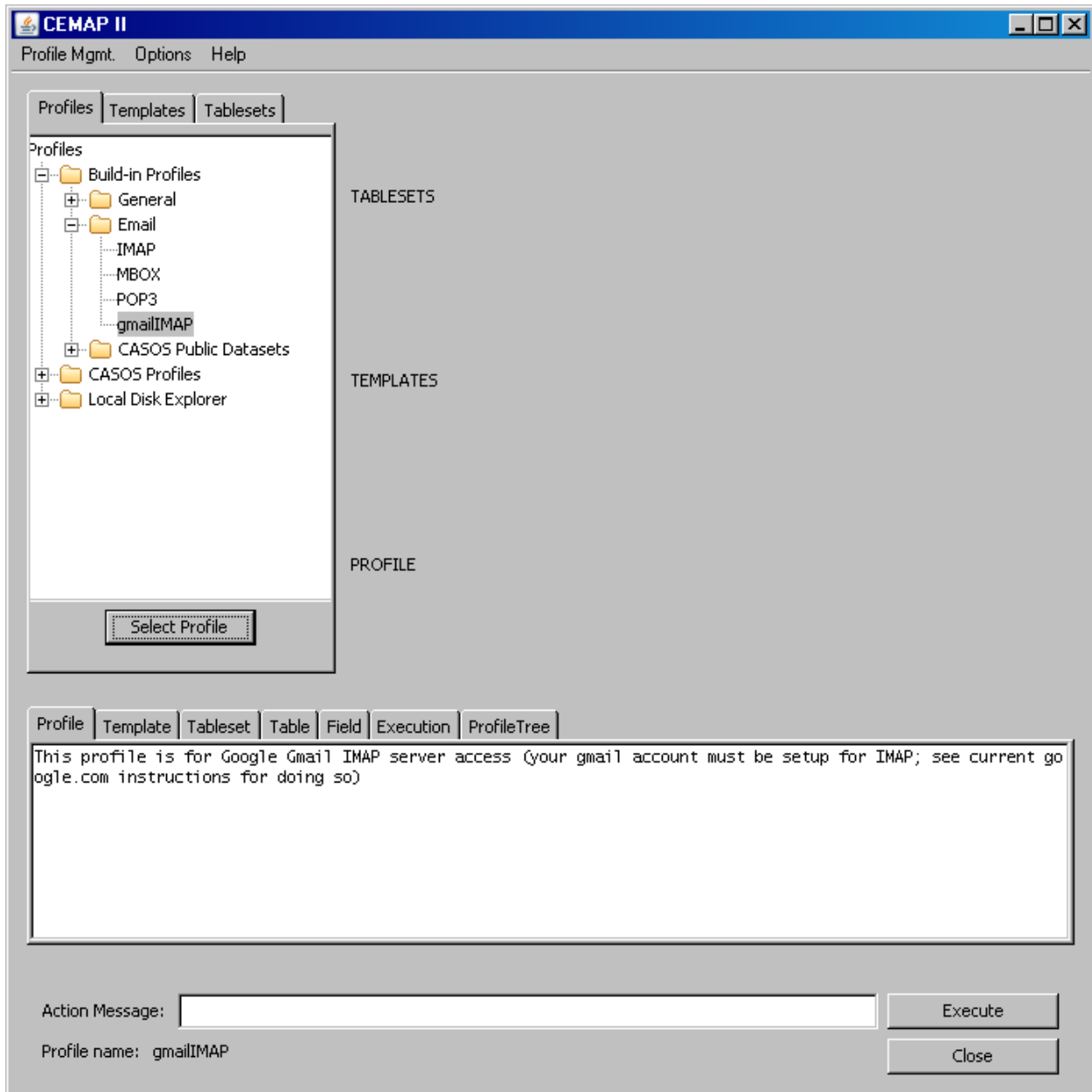


Figure 18. Result of Step 4.

Step 5:

Task: Change the output template to remove an unwanted network from the output, part I

Action: Left click the Templates tab in the top left window

Result: The template settings will show in the window, e.g. Fig. 19

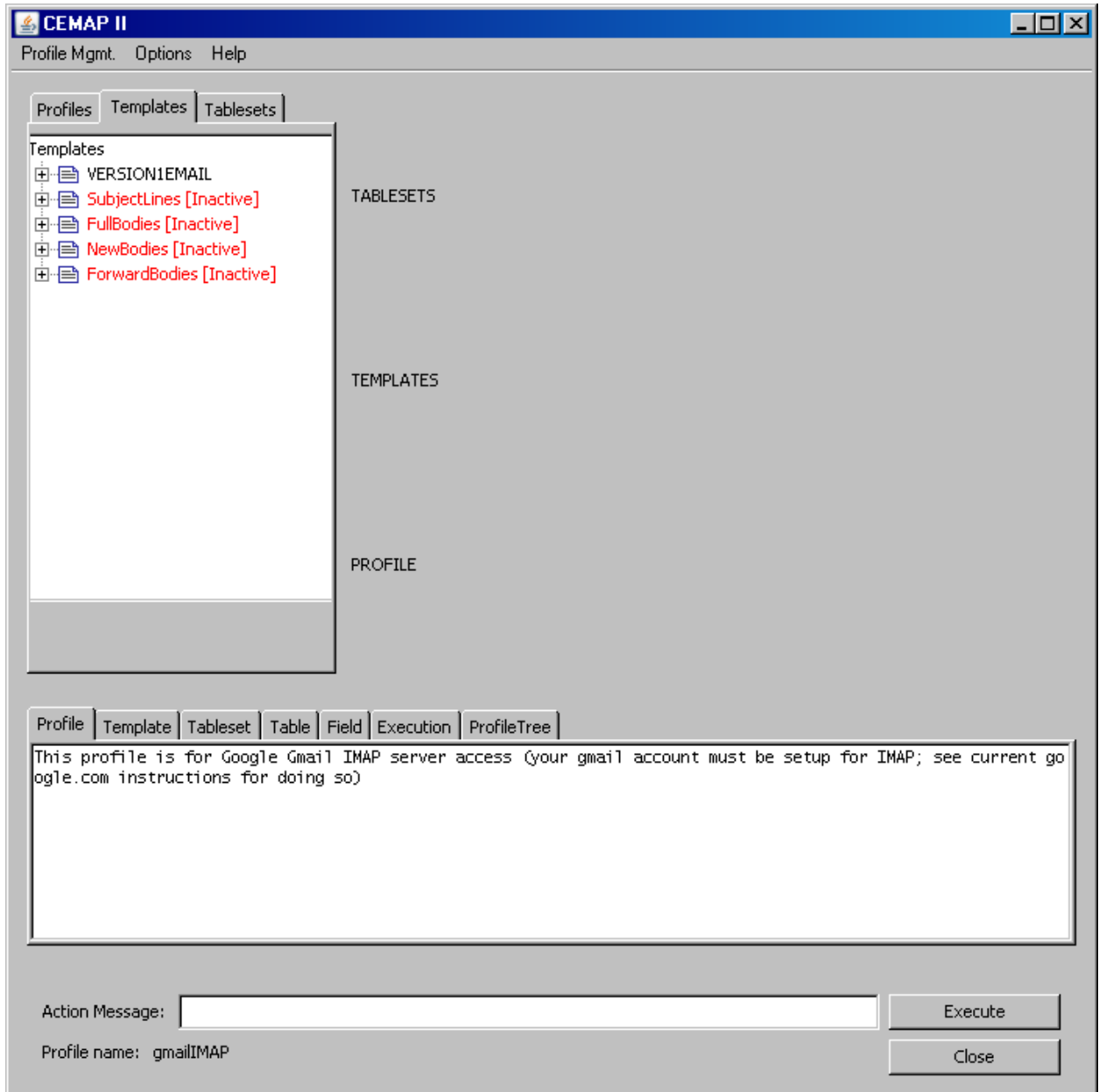


Figure 19. Result of Step 5.

Step 6:

Task: Change the output template to remove an unwanted network from the output, part II, expand the VERSION1EMAIL item

Action: Left click the + symbol in front of VERSION1EMAIL

Result: The template window will show the network template name: IMAP Profile, e.g. Fig. 20

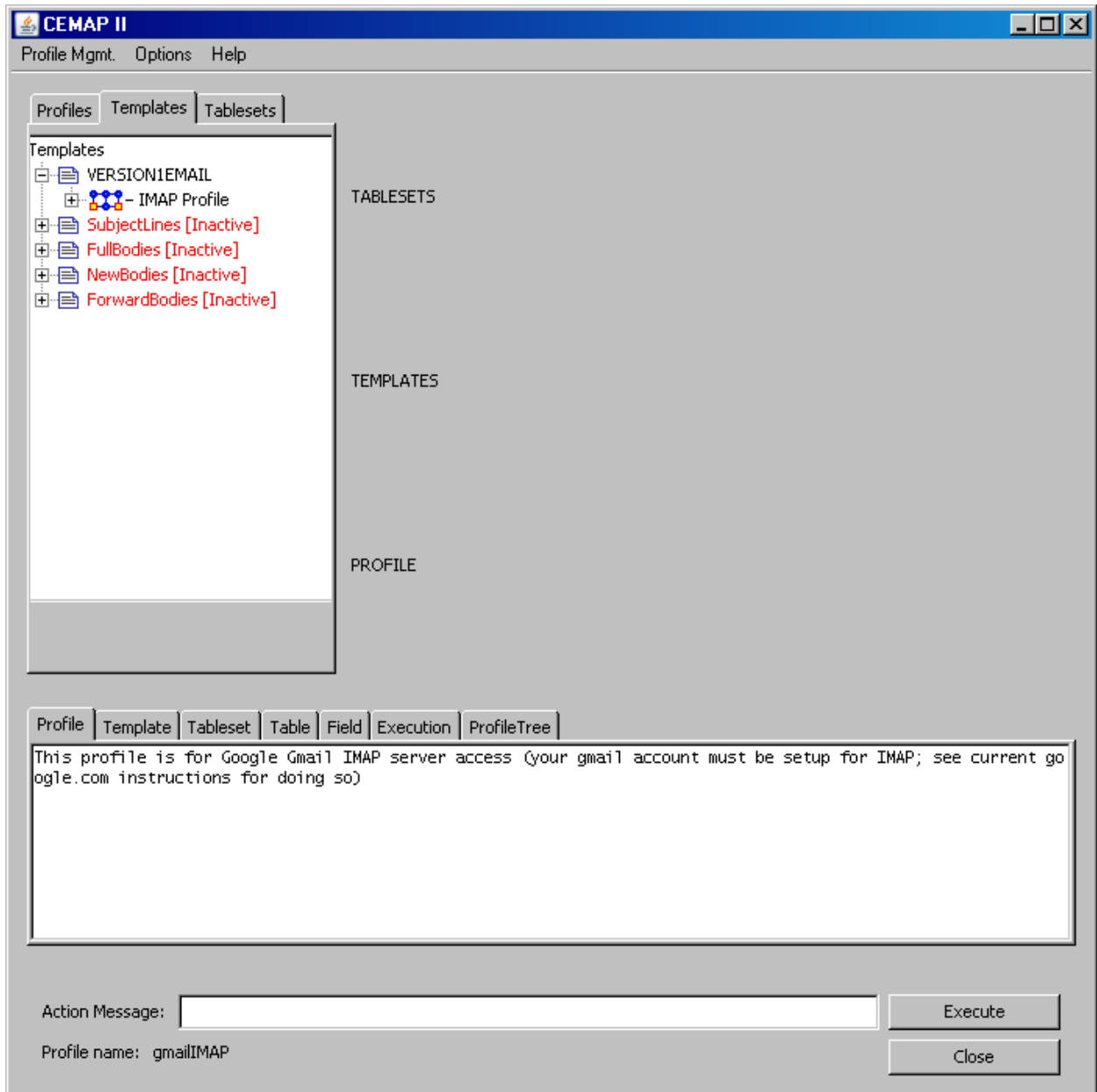


Figure 20. Result of Step 6.

Step 7:

Task: Change the output template to remove an unwanted network from the output, part III, expand the IMAP Profile item

Action: Left click the + symbol in front of IMAP Profile

Result: The template window will show the nodeclasses and networks that will be in the output, e.g. Fig. 21

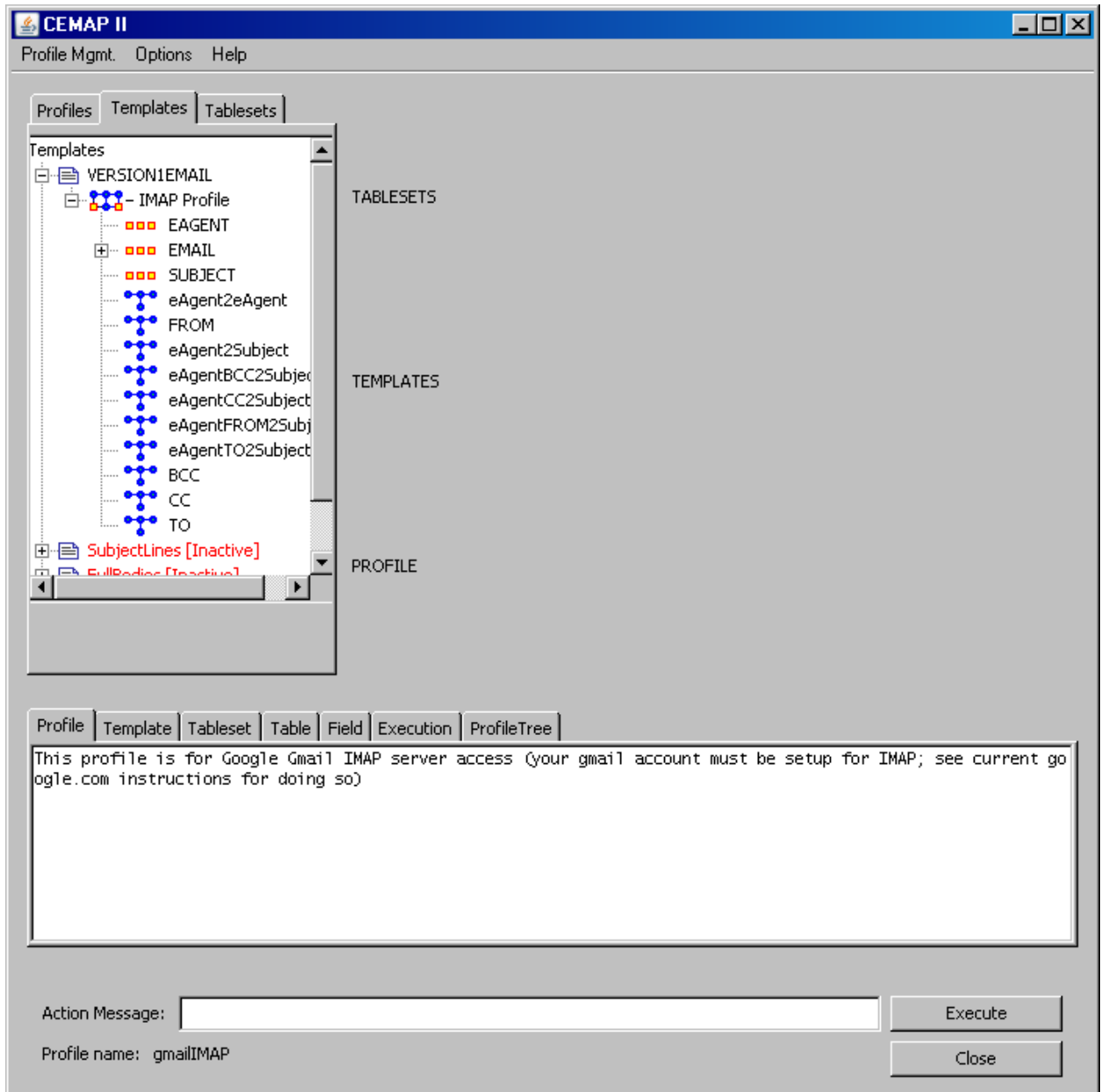


Figure 21. Result of Step 7.

Step 8:

Task: Change the output template to remove an unwanted network from the output, part IV, identify the network to exclude/remove

Action: Left click the FROM network item

Result: The delete option will show in the window, e.g. Fig. 22

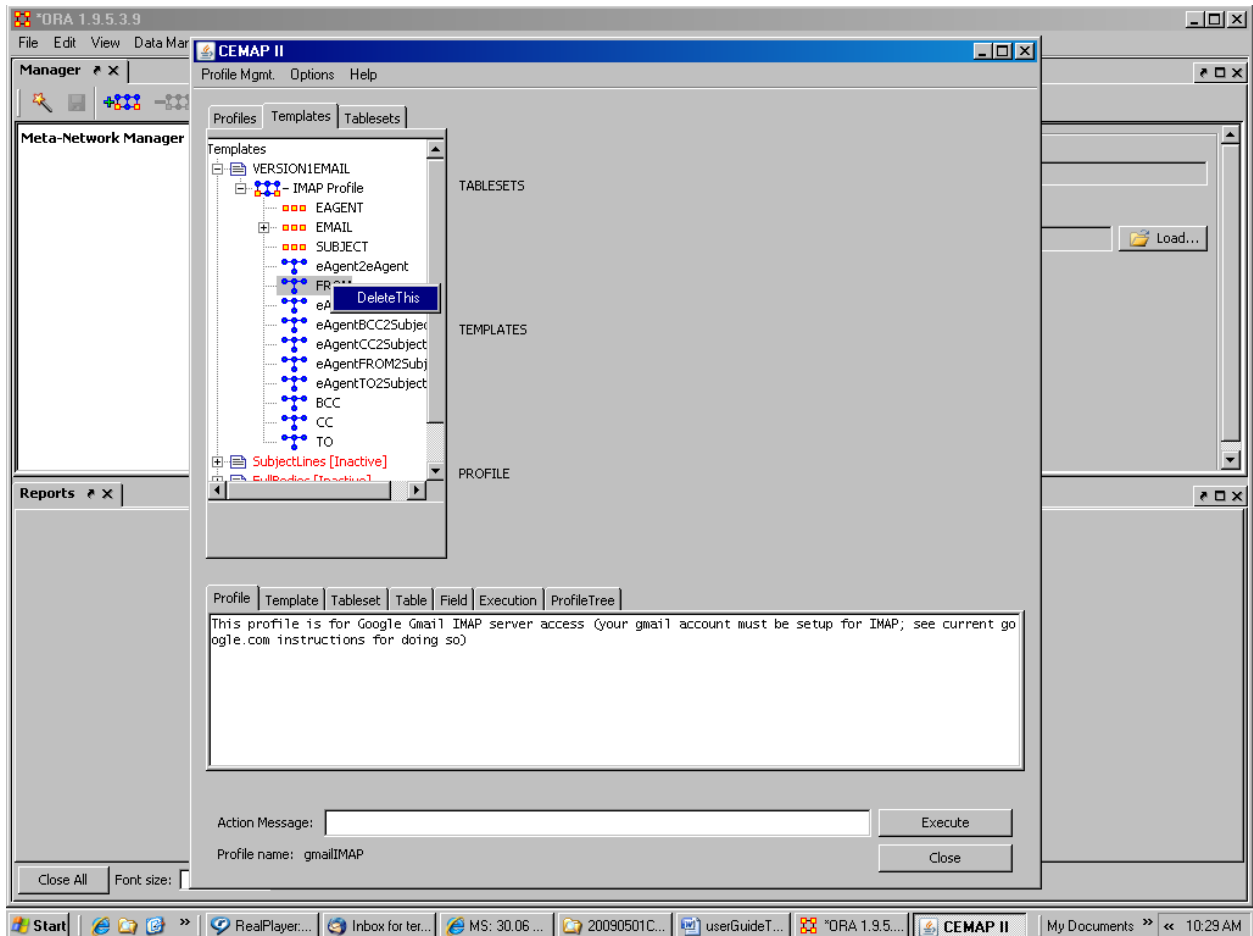


Figure 22. Result of Step 8.

Step 9:

Task: Change the output template to remove an unwanted network from the output, part V, now delete it!

Action: Right click the Delete option

Result: The FROM network will be removed from the IMAP Profile meta-network, e.g. Fig. 23

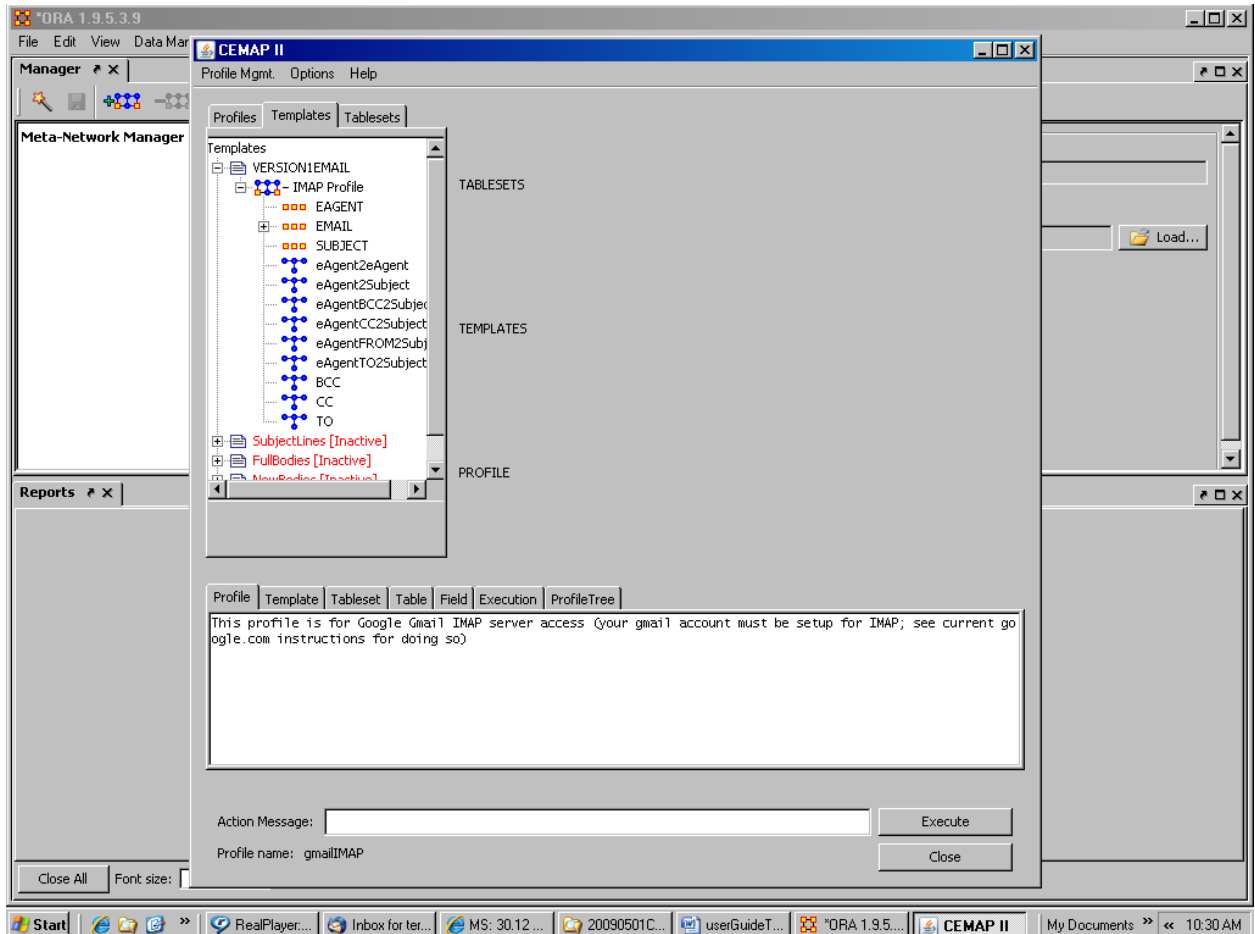


Figure 23. Result of Step 9.

Step 10:

Task: Execute the profile that is in memory

Action: Left Right click the Execute button (bottom right of the CEMAP window)

Result: The template fields window will be displayed (this is where the operator can complete any template fields, like in this case, the destination of the network file, e.g. Fig. 24

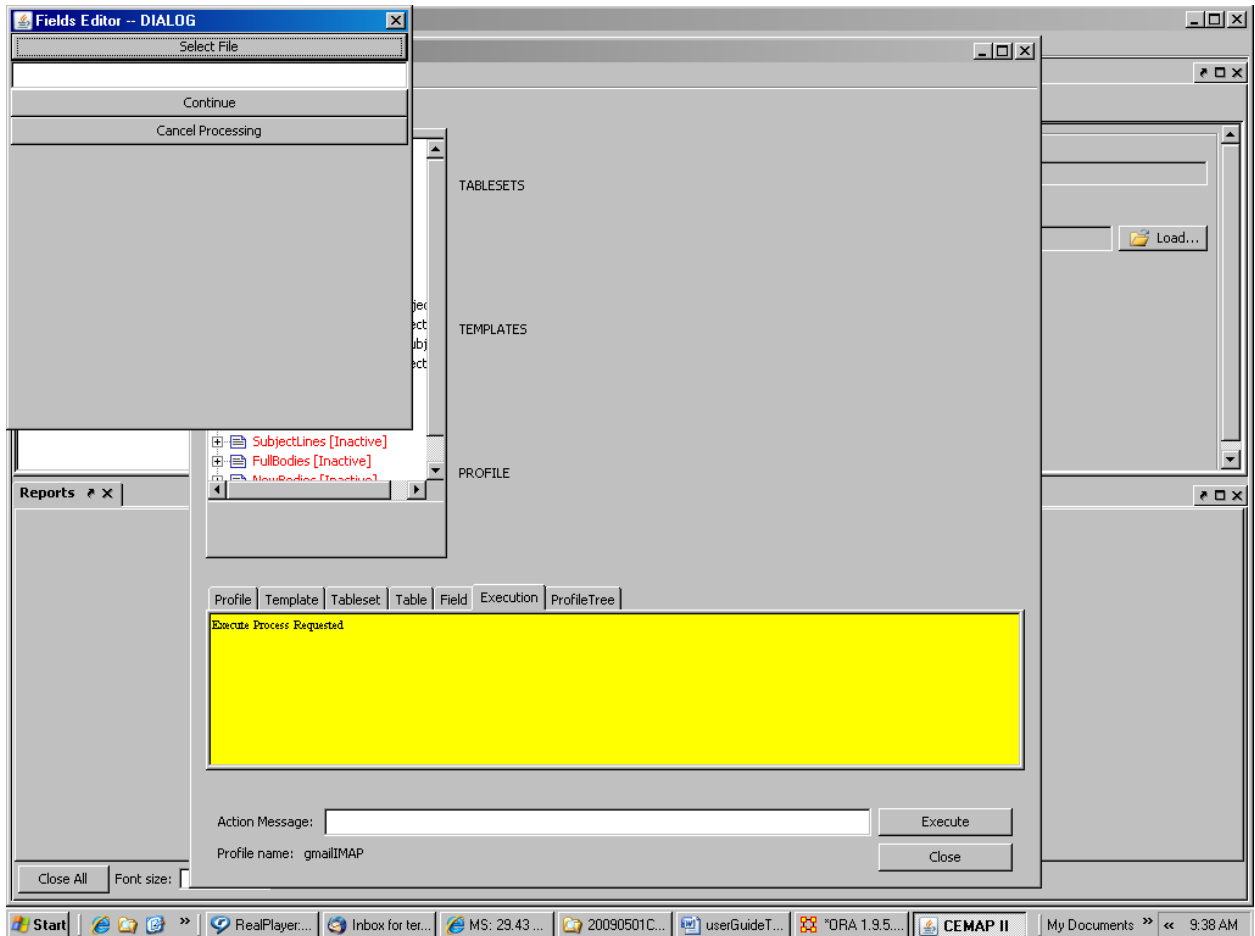


Figure 24. Result of Step 10.

Step 11:

Task: Skip over the Template Fields input window

Action: Right click the continue button shown in the Template Fields window

Result: The tableset fields window will be displayed (this is where the operator can complete any tableset fields, like in this case, the values necessary for an email server, e.g. Fig. 25

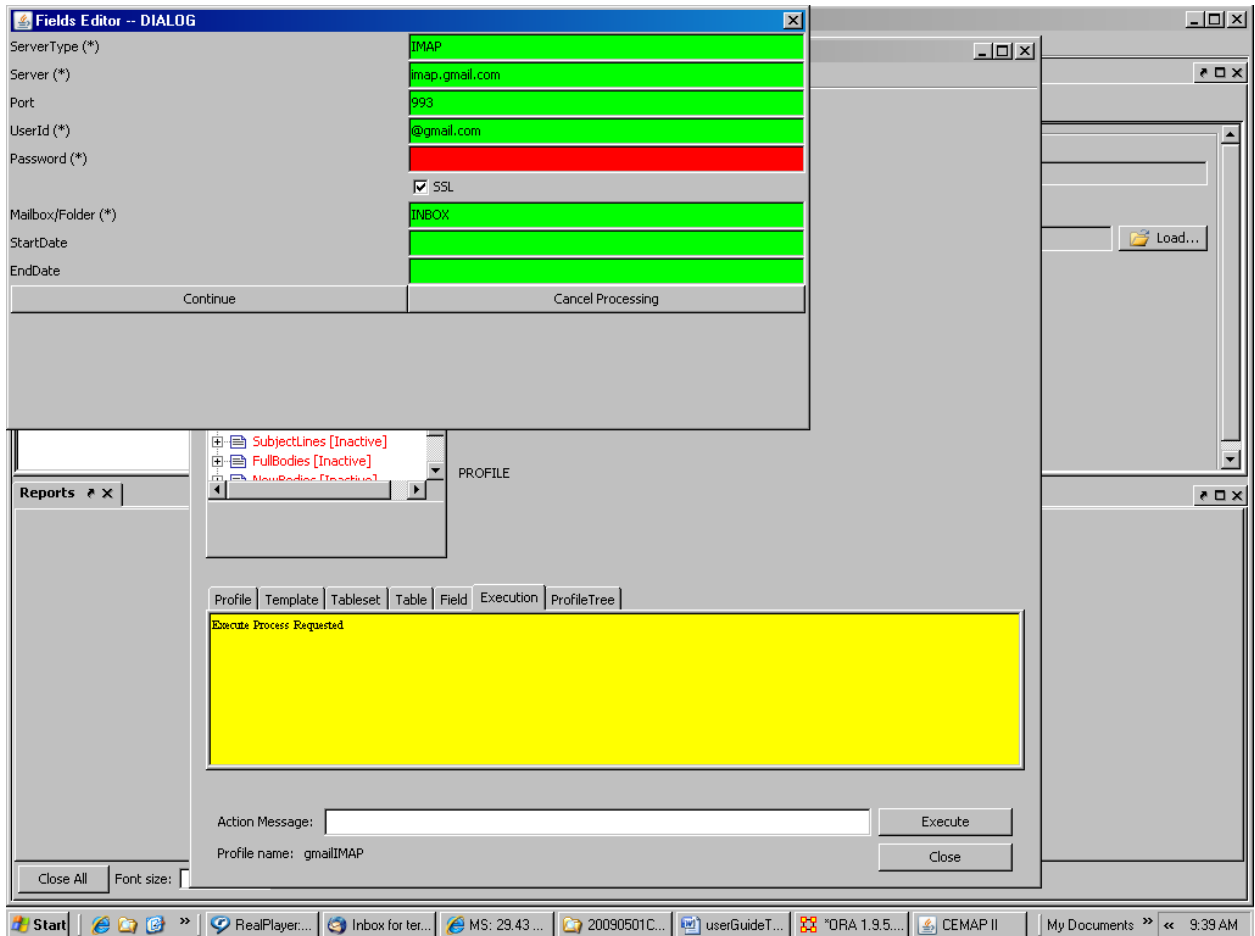


Figure 25. Result of Step 11.

Step 12:

Task: Provide the email users id

Action: Click to the UserId field and enter your user Id for GMAIL

Result: The field value settings will show in the window, e.g. Fig. 26

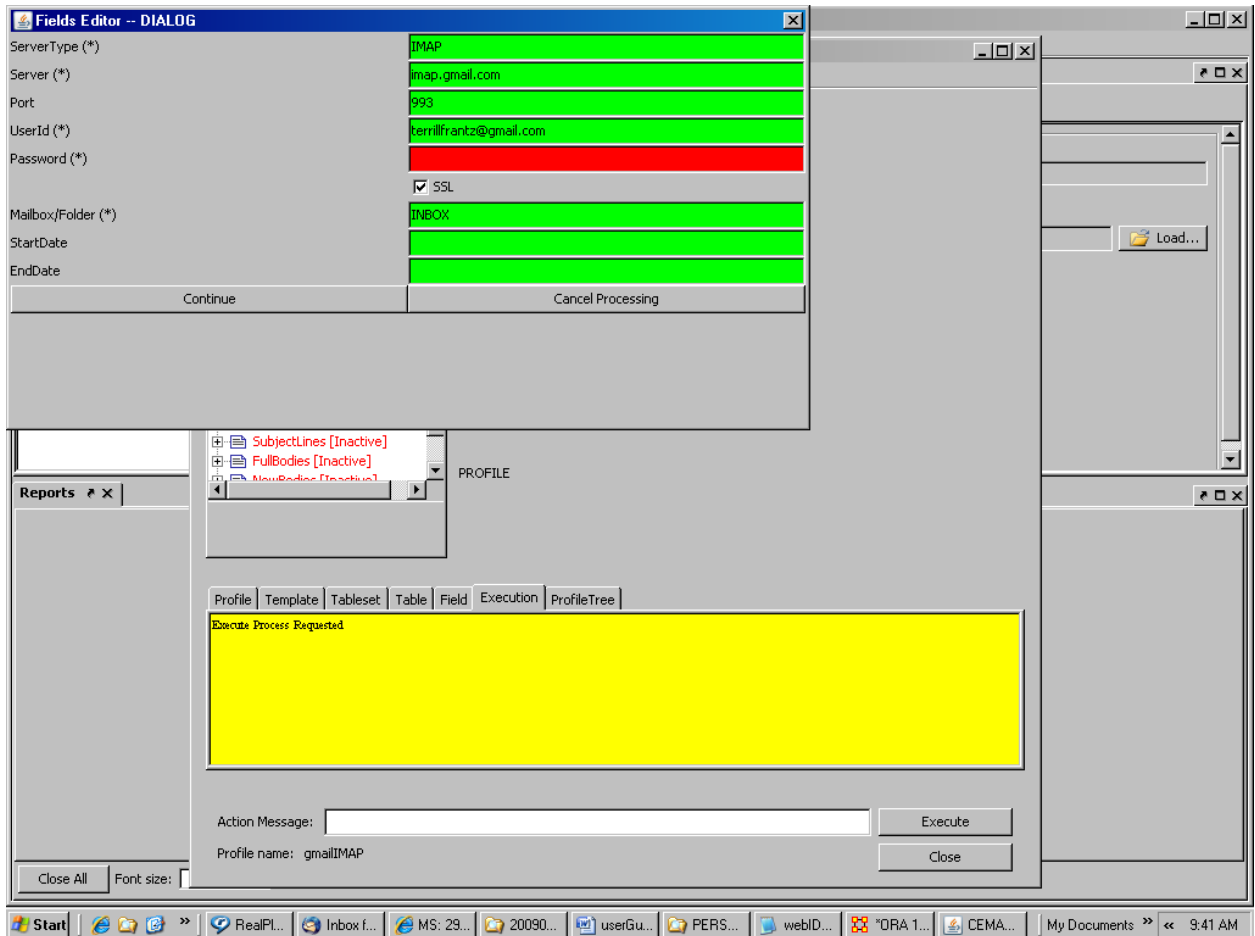


Figure 26. Result of Step 12.

Step 13:

Task: Enter your email server user password

Action: Click or tab to the Password field and enter your user password for GMAIL

Result: The template settings will show in the window, e.g. Fig. 27

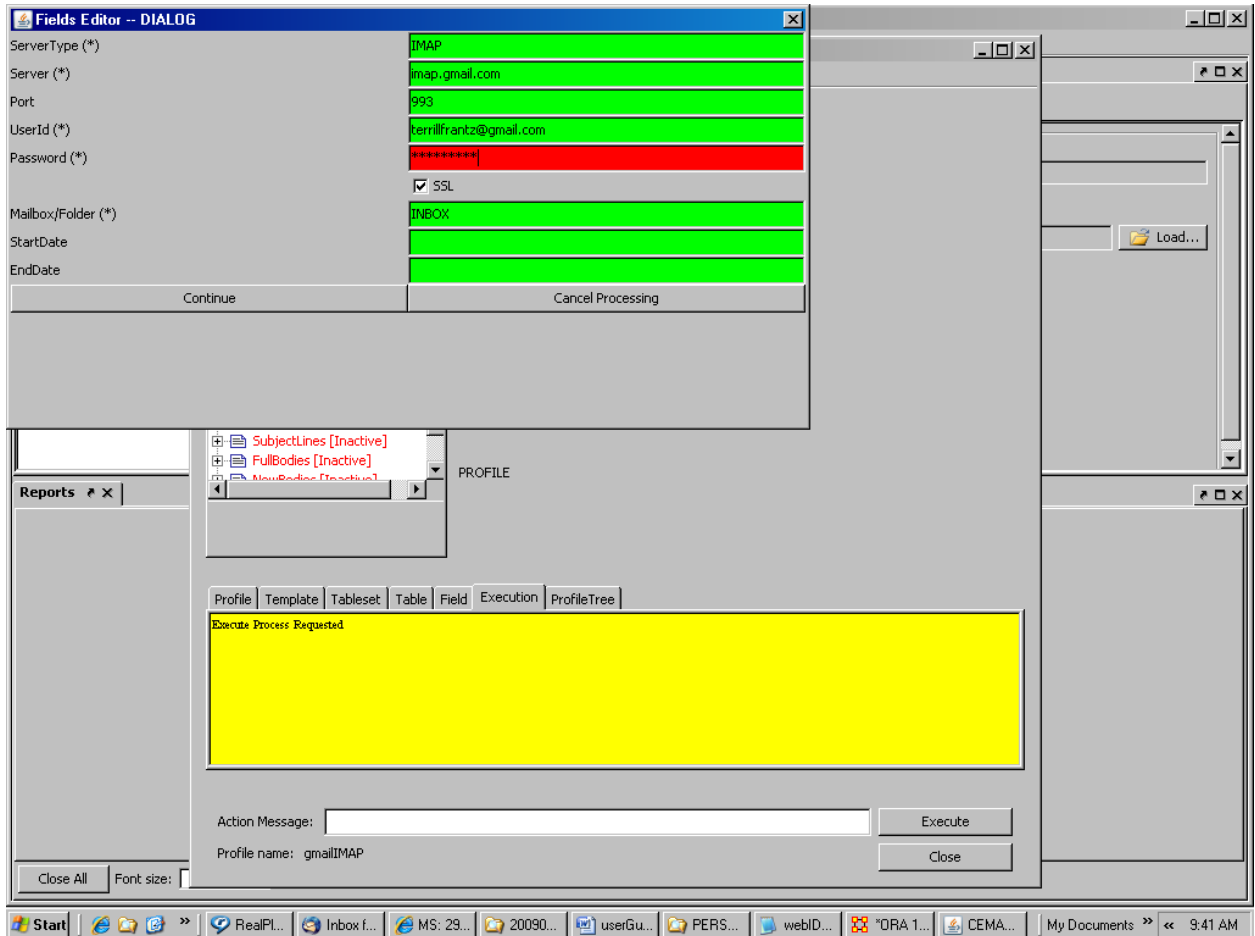


Figure 27. Result of Step 13.

Step 14:

Task: Complete the Tableset fields input and continue through to execute the profile

Action: Right click the Continue button

Result: A progress window will appear and will count the emails as they are processed from the email server, e.g. Fig. 28

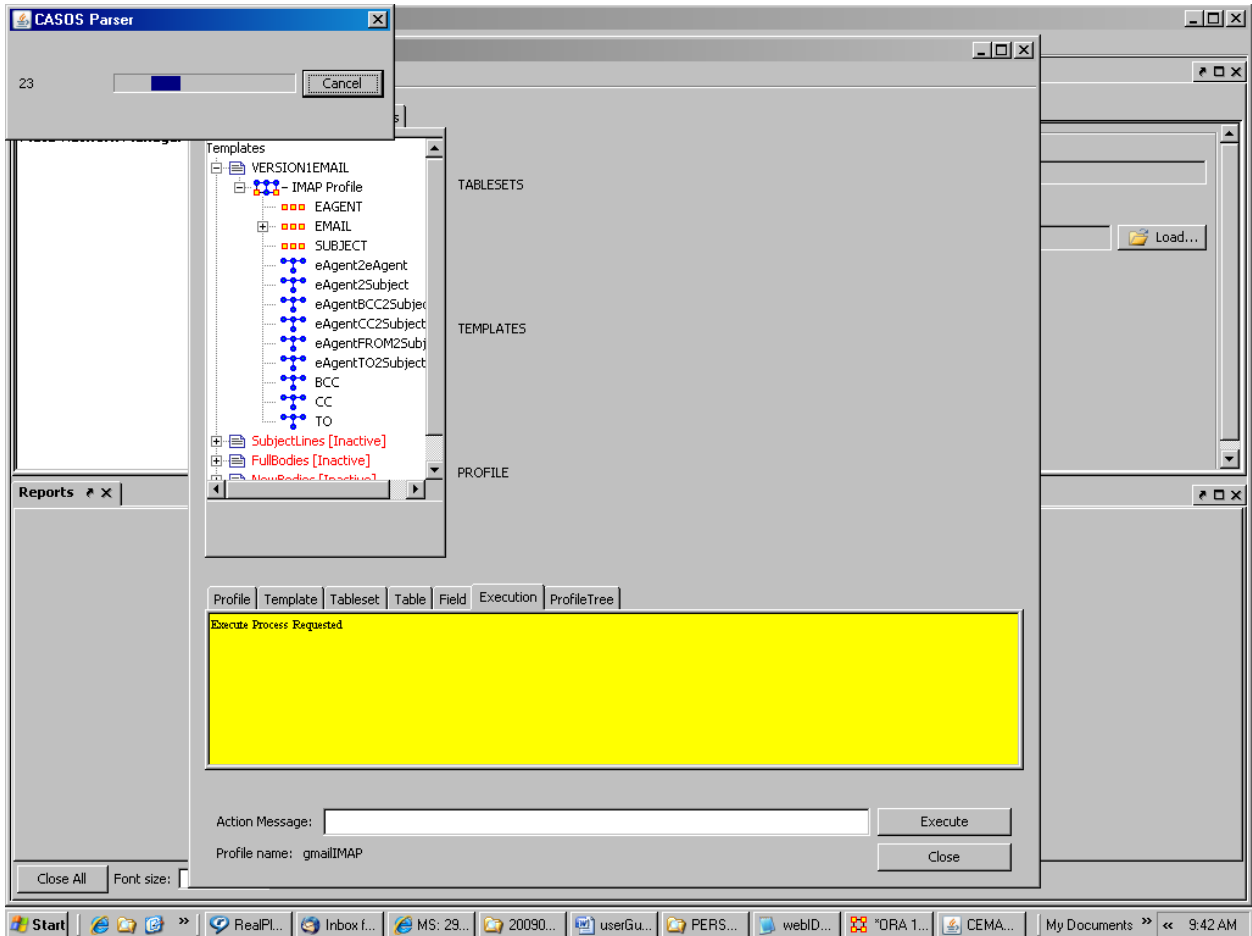


Figure 28. Result of Step 14.

Step 15:

Task: Wait for the emails to all process

Action: Wait ... until the Parsing Complete window appears

Result: The Parsing Complete message window will appear, e.g. Fig. 29

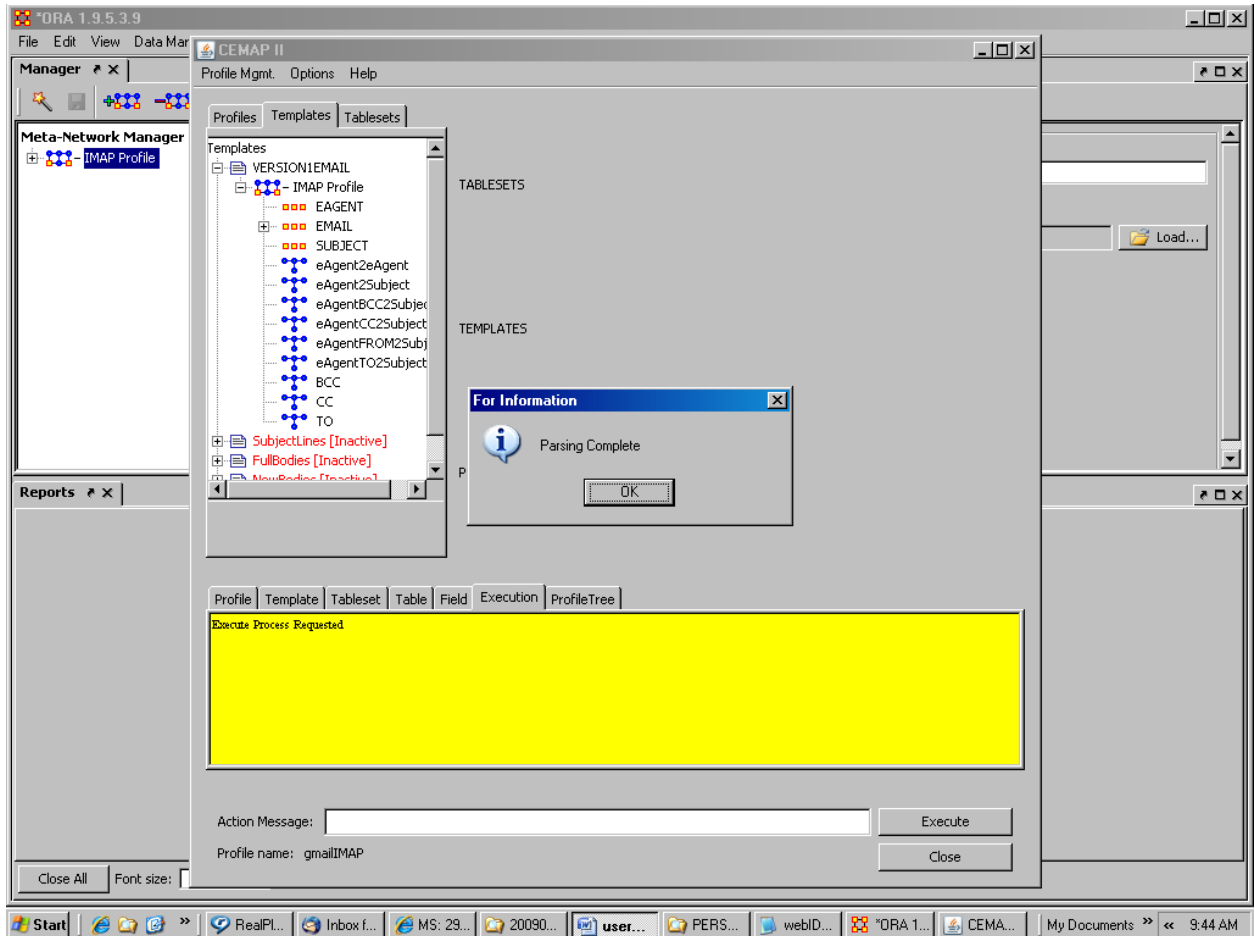


Figure 29. Result of Step 15.

Step 16:

Task: Acknowledge completion of the email processing

Action: Right click the OK button on the Parser Complete message window

Result: The Parsing Complete message window will disappear, e.g. Fig. 30

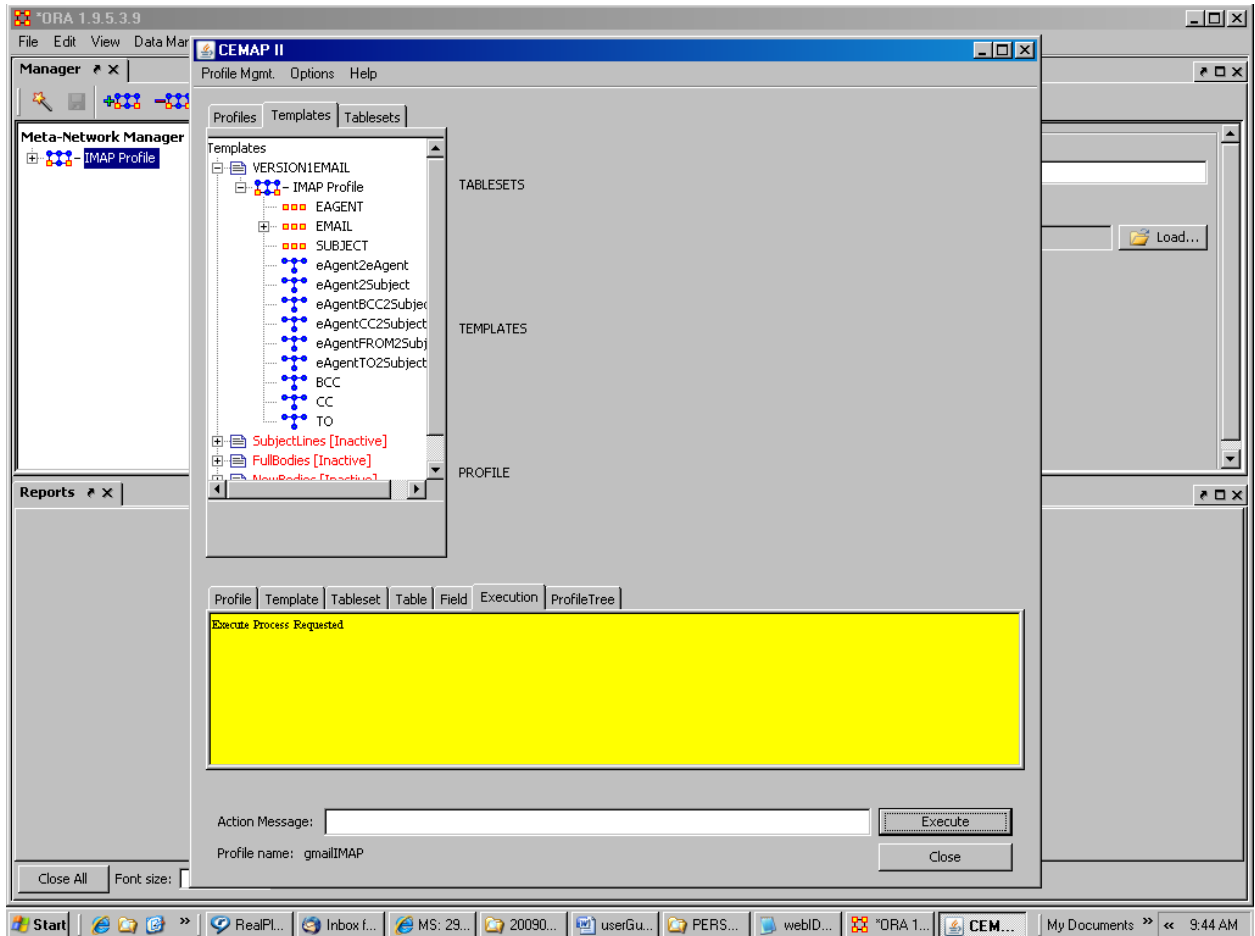


Figure 30. Result of Step 16.

Step 17:

Task: Take a look at the email data in ORA

Action: Left click the IMAP Profile meta-network in the ORA Window

Result: The CEMAP window will go behind the ORA window and the IMAP Profile meta-network item in the ORA data tree will be highlighted, e.g. Fig. 31

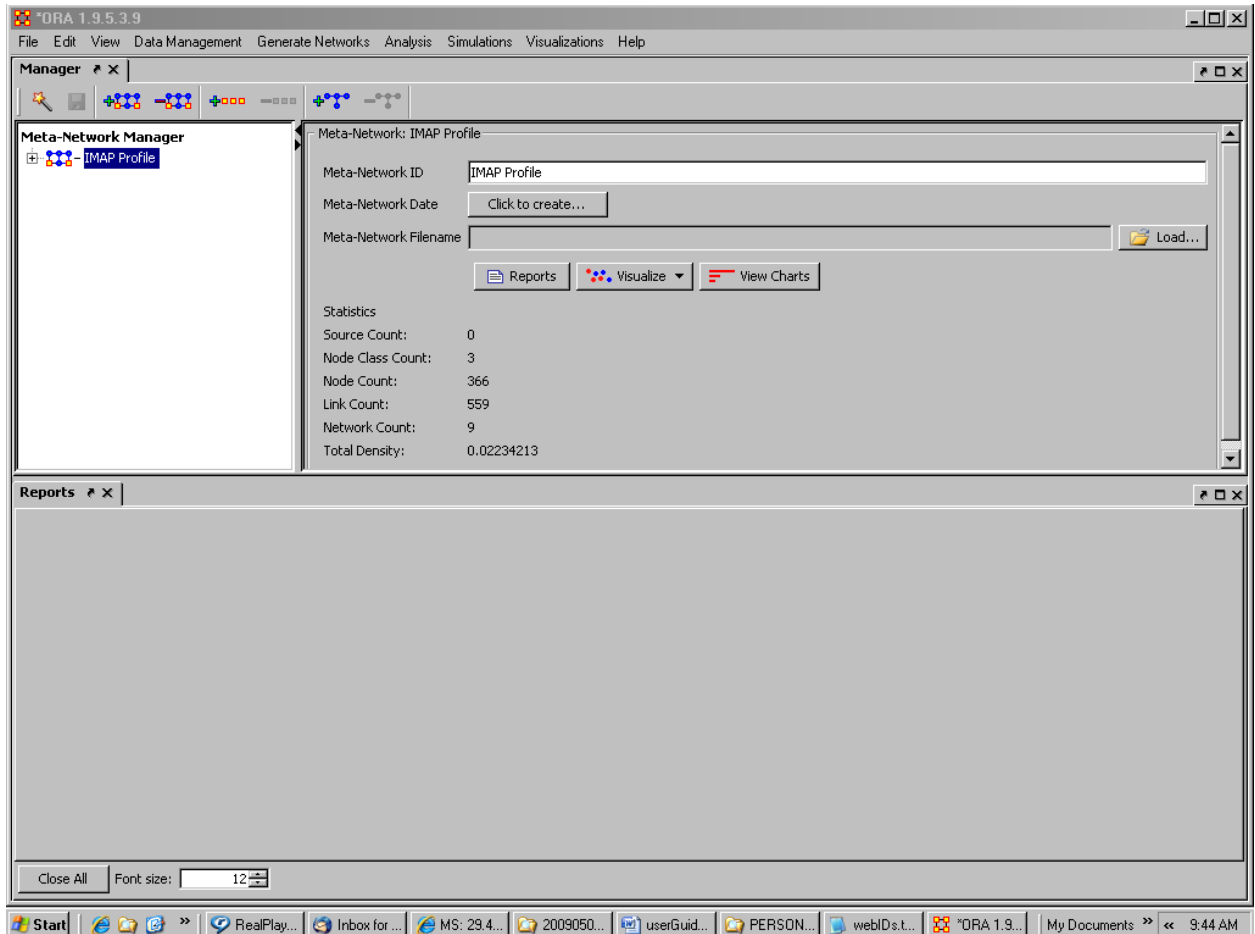


Figure 31. Result of Step 17.

Step 18:

Task: Expand the meta-network to explore the data

Action: Left click the “+” in front of the IMAP Profile item

Result: The meta-network tree will expand, e.g. Fig. 32

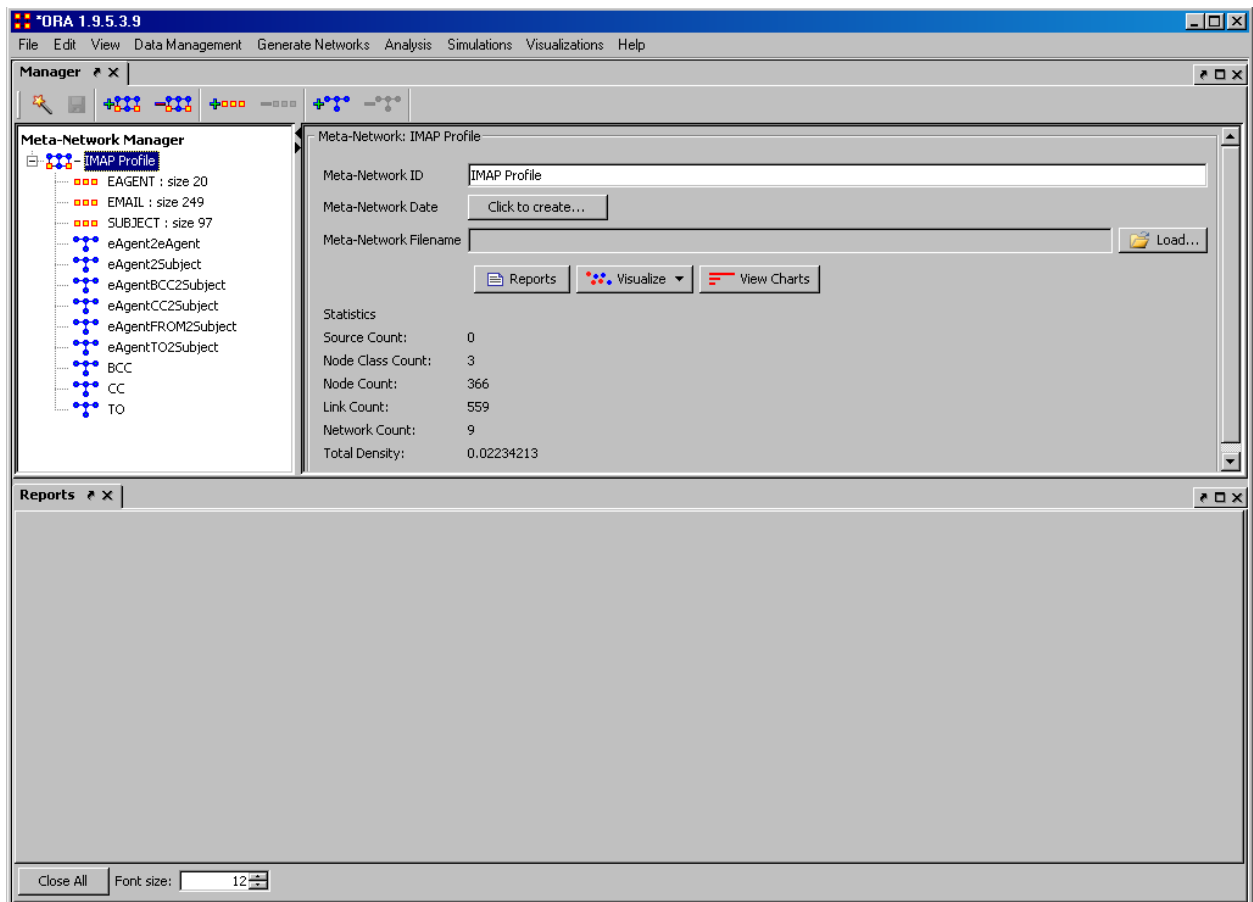


Figure 32. Result of Step 18.

Step 19:

Task: Return to the CEMAP window

Action: Use the windows toolbar to bring up the CEMAP window

Result: The CEMAP window will be in front of the ORA window, e.g. Fig. 33

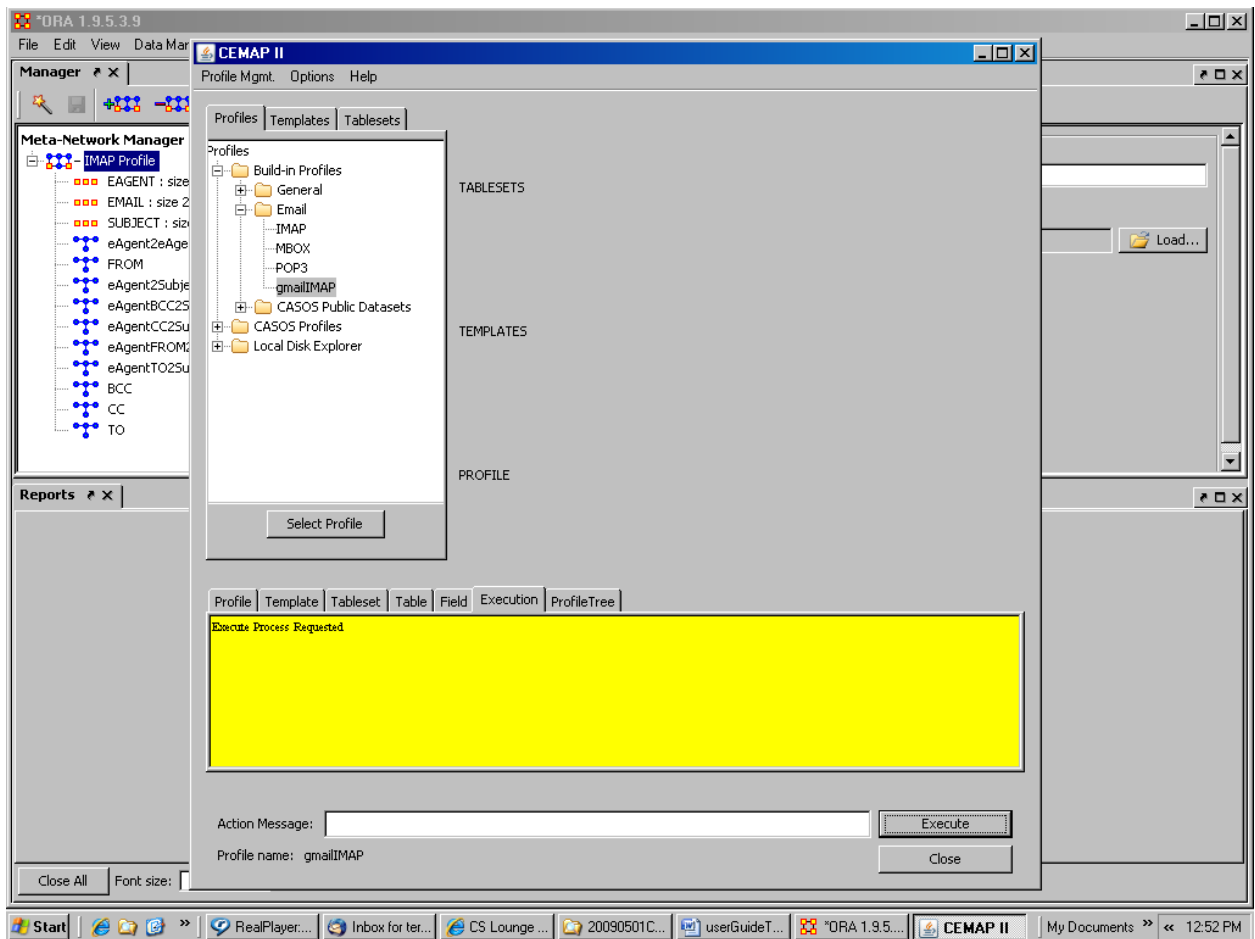


Figure 33. Result of Step 19.

Step 20:

Task: Go to the save the active profile menu option

Action: Left click the Profile Mgmt option on the top of the CEMAP window

Result: The Profile Mgmt menu will appear, e.g. Fig. 34

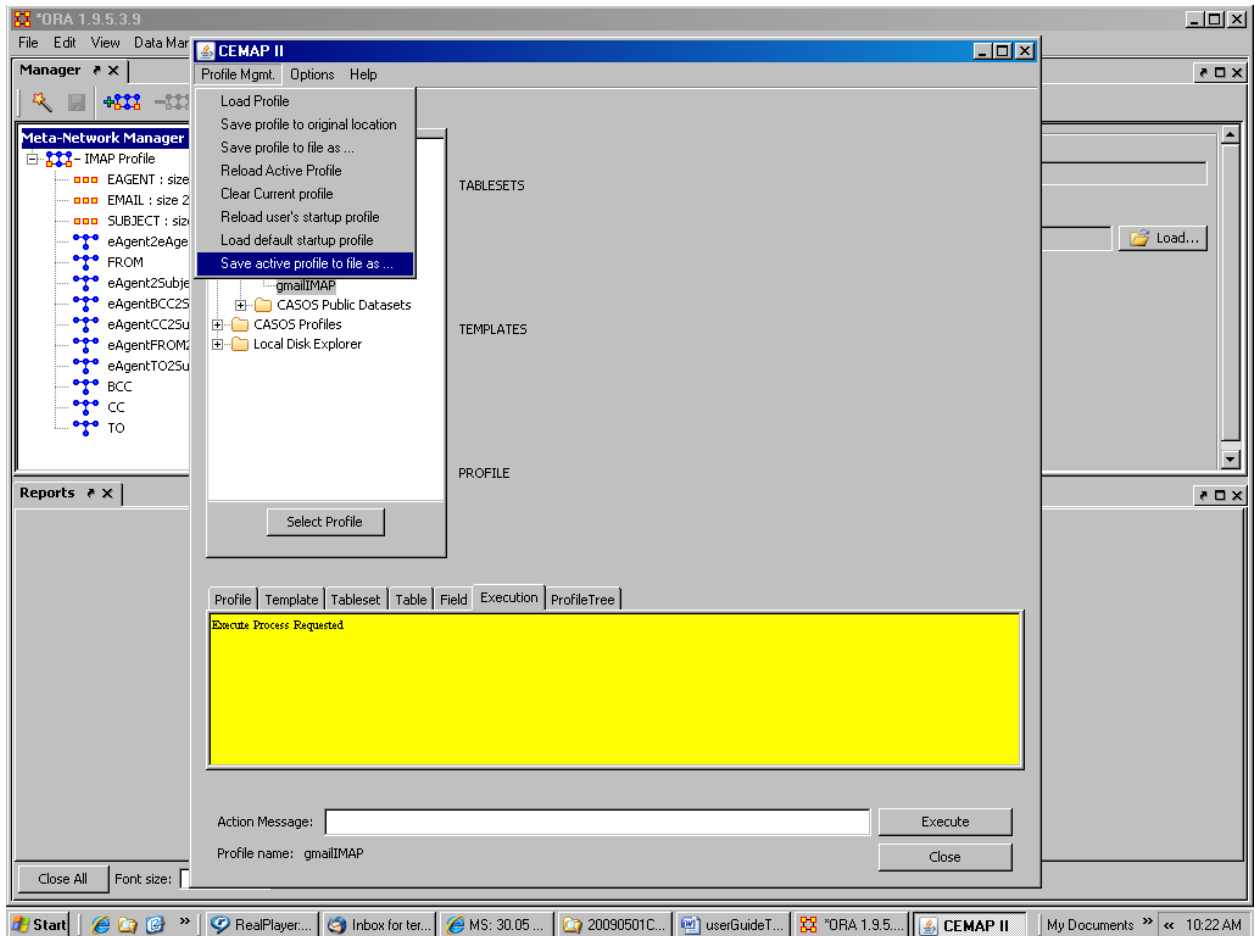


Figure 34. Result of Step 20.

Step 21:

Task: Select the “Save active profile as...” option

Action: Right click the “Save active profile as...” option

Result: The Windows File explorer window will open, e.g. Fig. 35

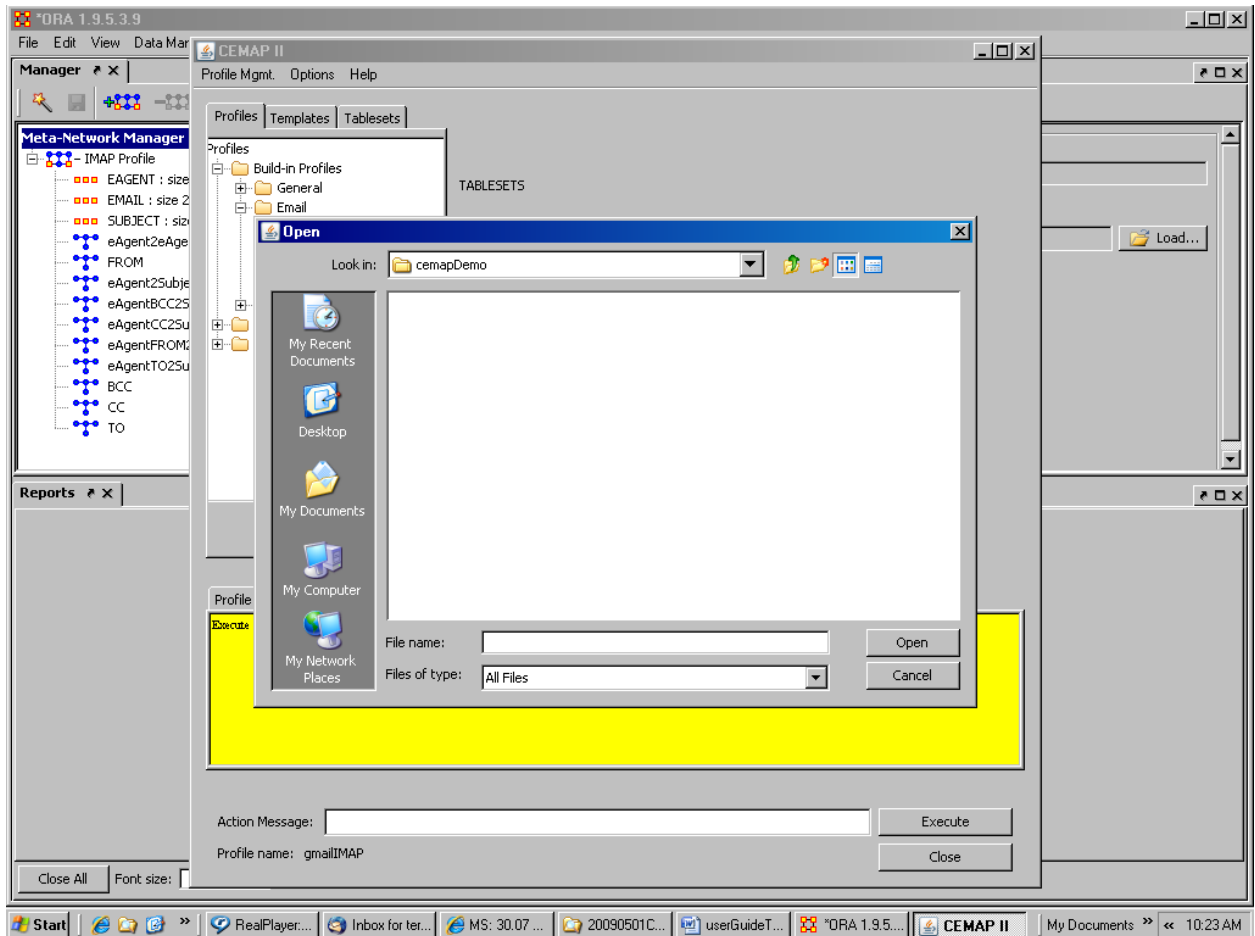


Figure 35. Result of Step 21.

Step 22:

Task: Entering in the file name of where the profile is to be saved

Action: Type in the filename – suggest using the “.xml” filename extension—but not necessary

Result: The filename will appear in the file explorer window, e.g. Fig. 36

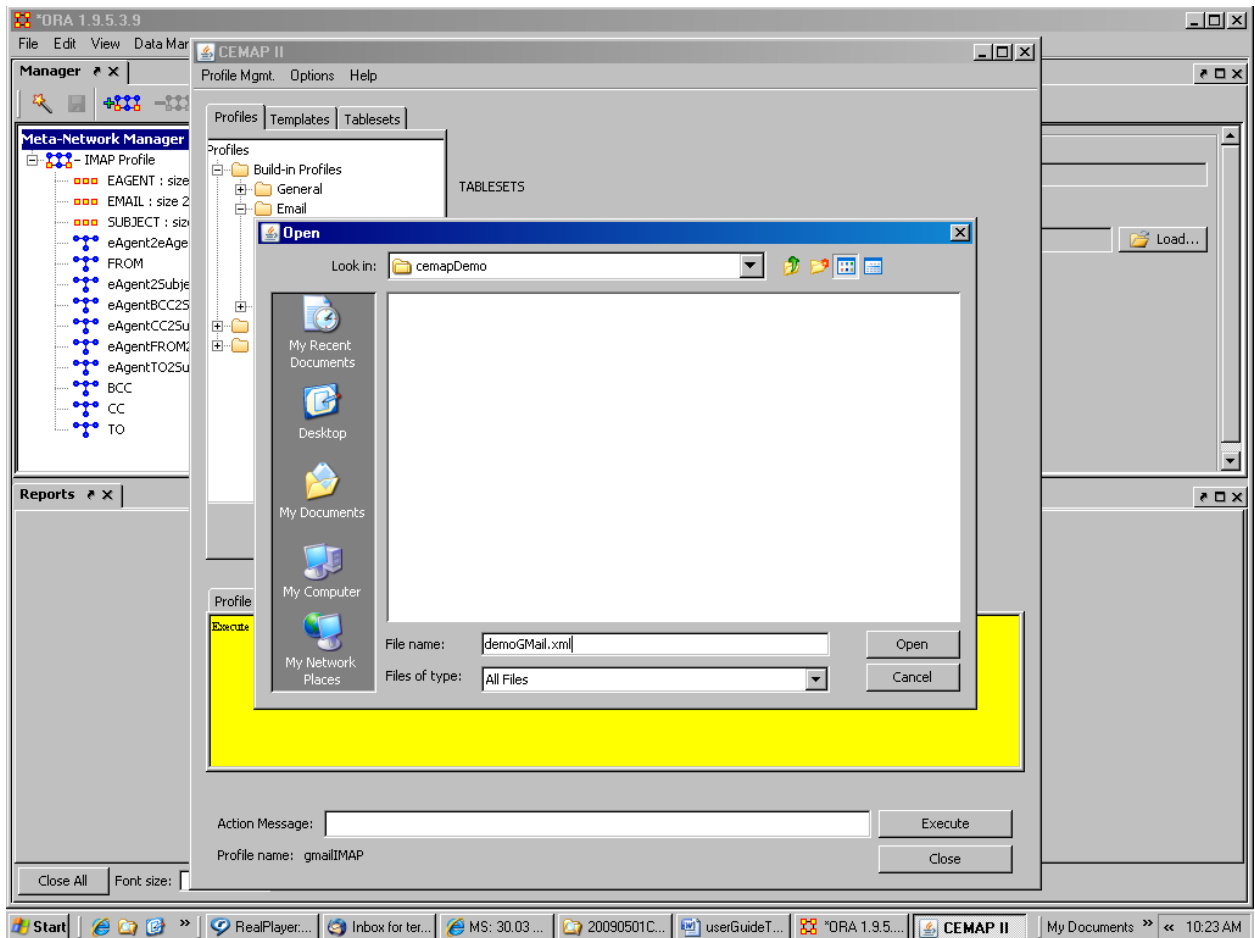


Figure 36. Result of Step 22.

Step 23:

Task: Complete the file naming process

Action: Right click the Open button

Result: The file explorer window will close and the profile is now saved, e.g. Fig.

37

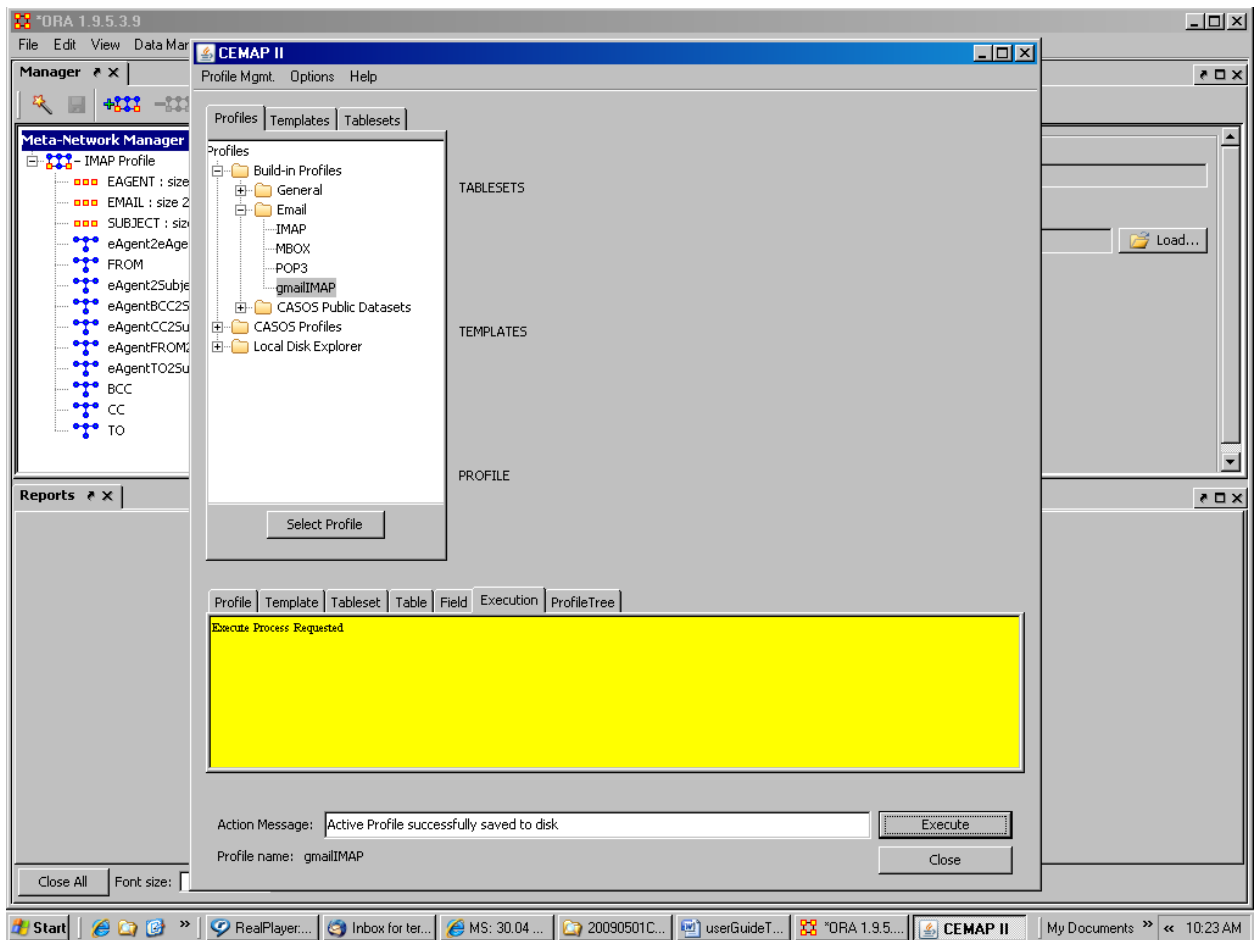


Figure 37. Result of Step 23.

Step 24:

Task: Load the saved profile

Action: Left click the Local Disk Explorer item in the cemap profile tree

Result: The file explorer window will appear, e.g. Fig. 38

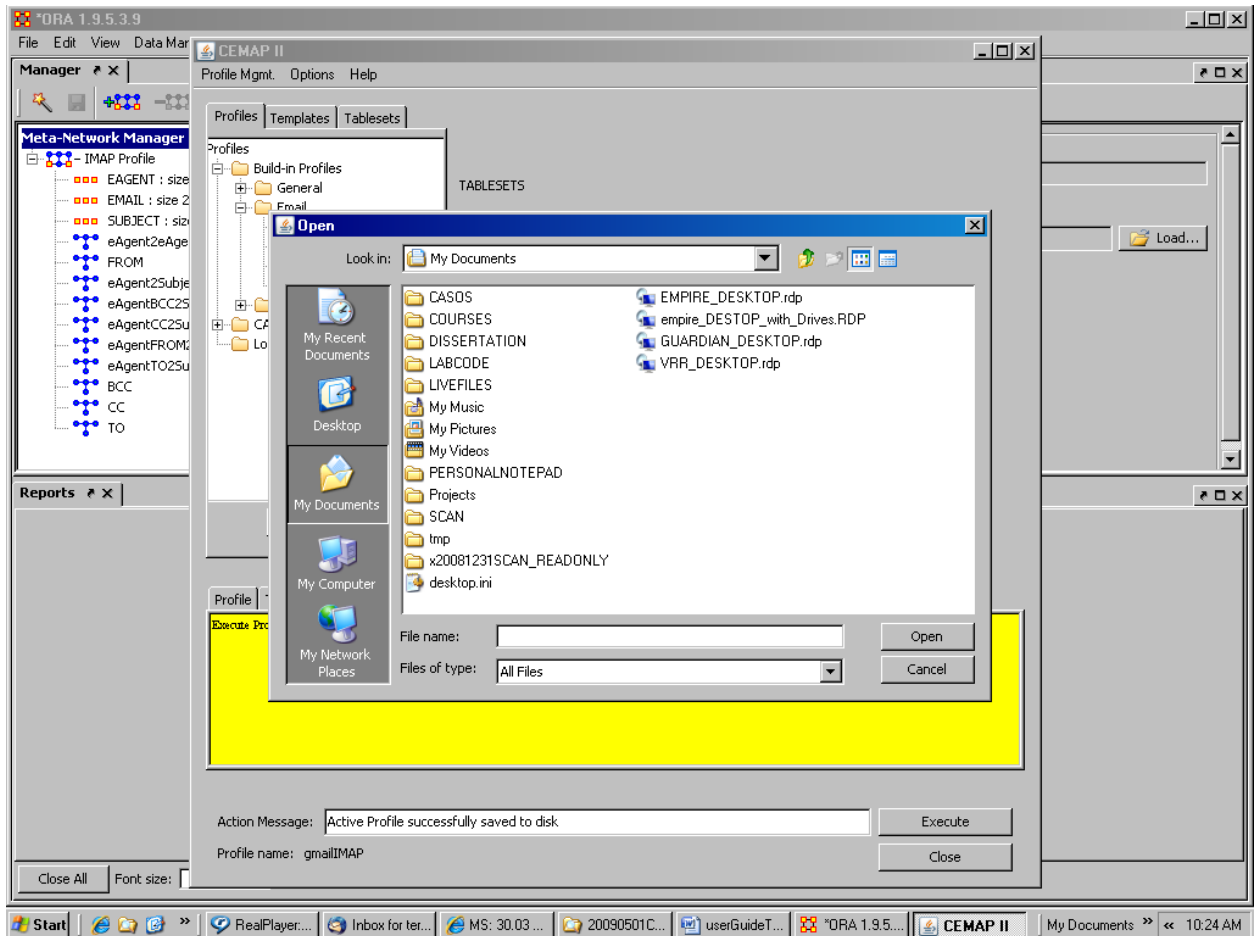


Figure 38. Result of Step 24.

Step 25:

Task: Locate the saved profile file

Action: Navigate through the file explorer window to locate your saved profile file

Result: The save profile file name will appear in the file explorer window, e.g. Fig. 39

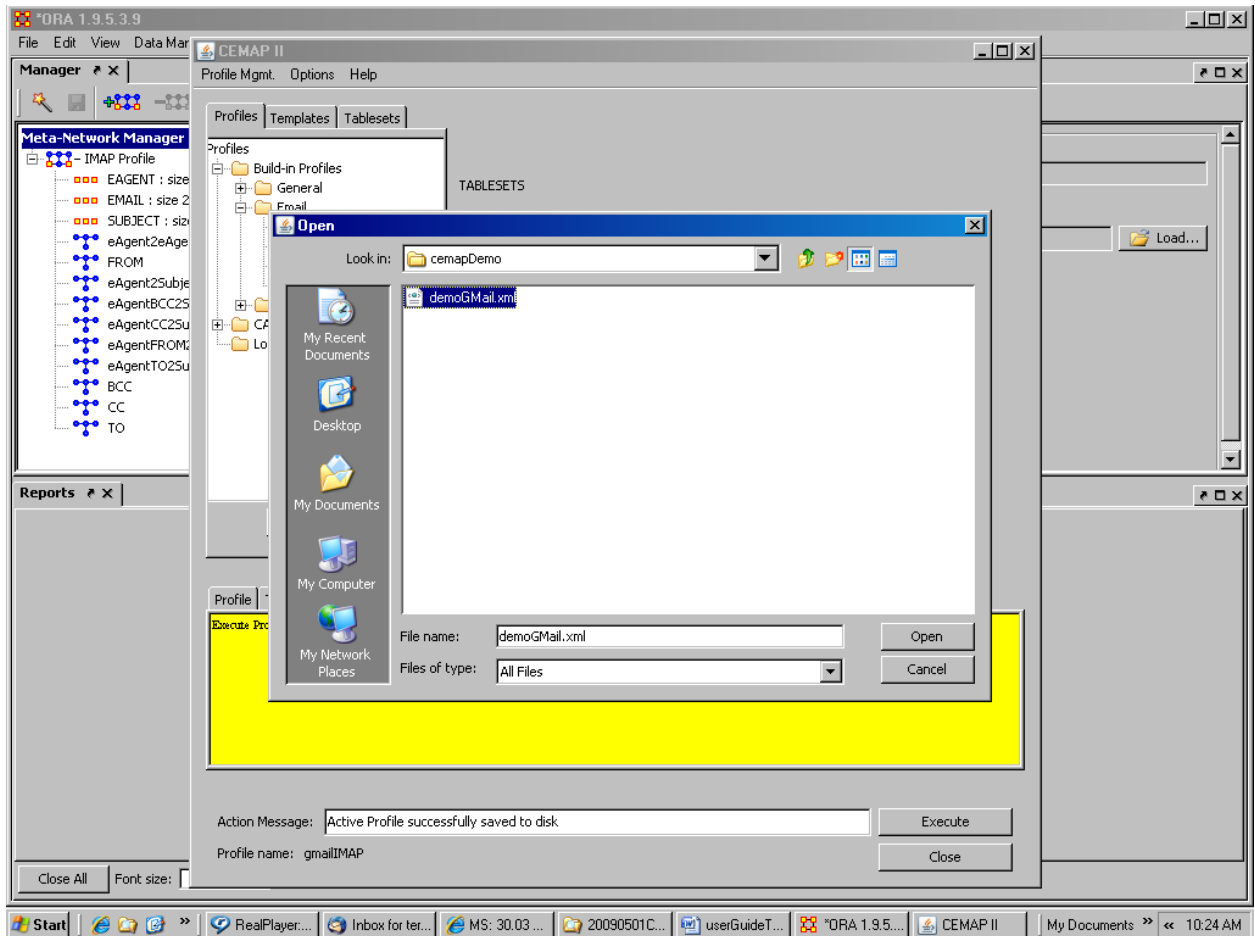


Figure 39. Result of Step 25.

Step 26:

Task: Select the saved profile filename

Action: Left click the Open button

Result: The profile will appear in the profiles tree, e.g. Fig. 40

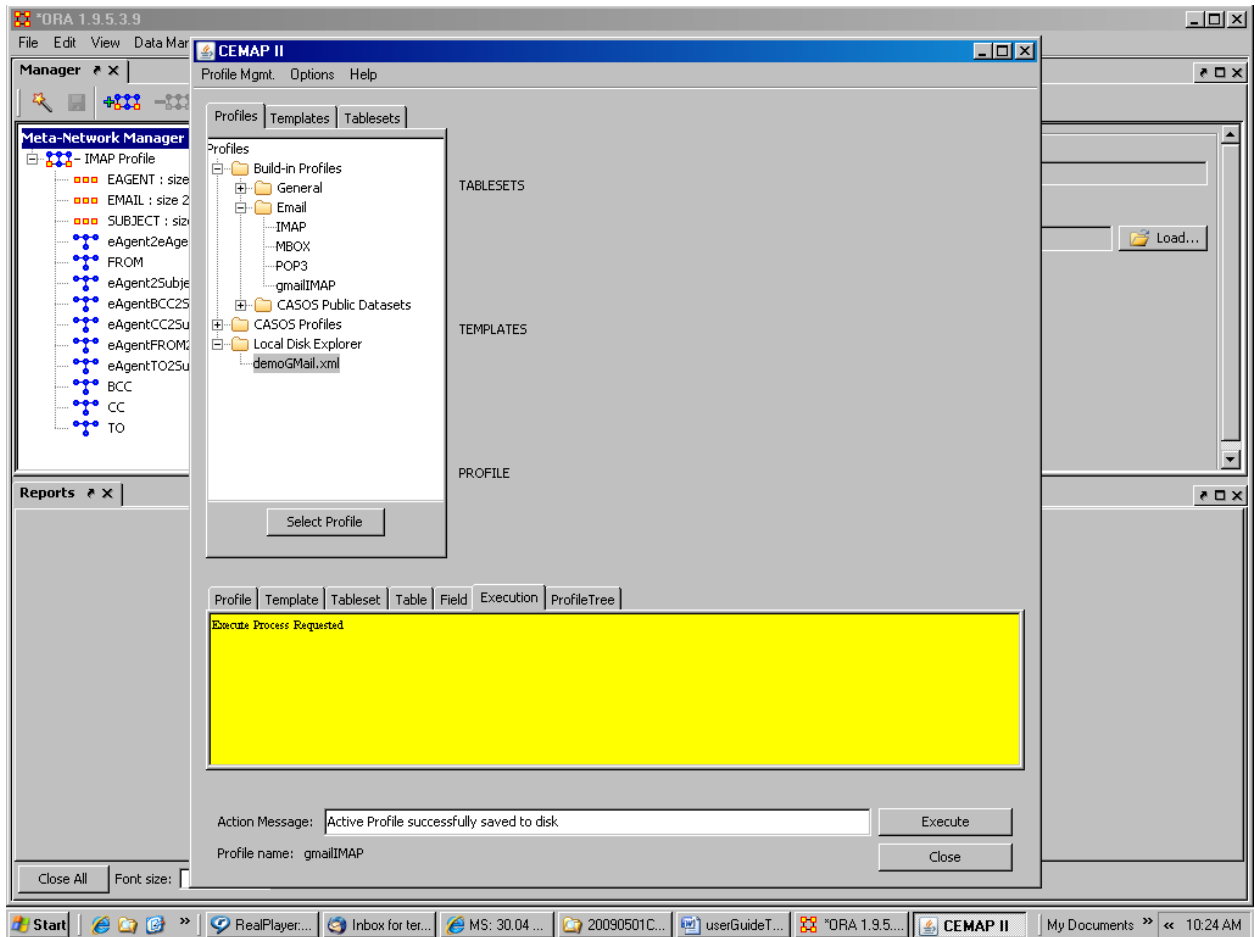


Figure 40. Result of Step 26.

Step 27:

Task: Select the saved profile file for loading

Action: Press the Select Profile button

Result: The profile will load and the Profile notes window will refresh , e.g. Fig.

41

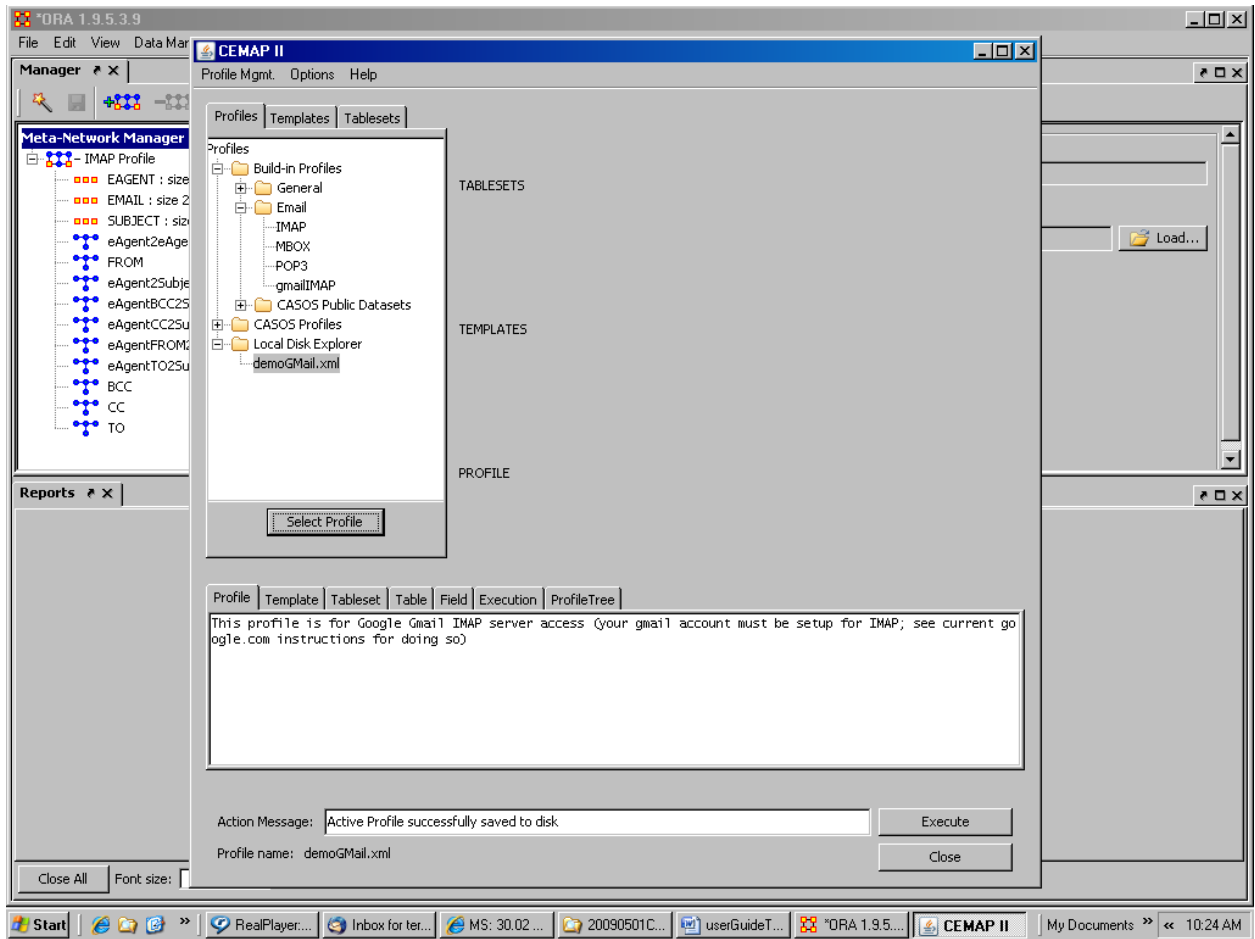


Figure 41. Result of Step 27.

Step 28:

Task: Execute the profile

Action: Left click the Templates Execute button

Result: The template fields widow will appear, e.g. Fig. 42

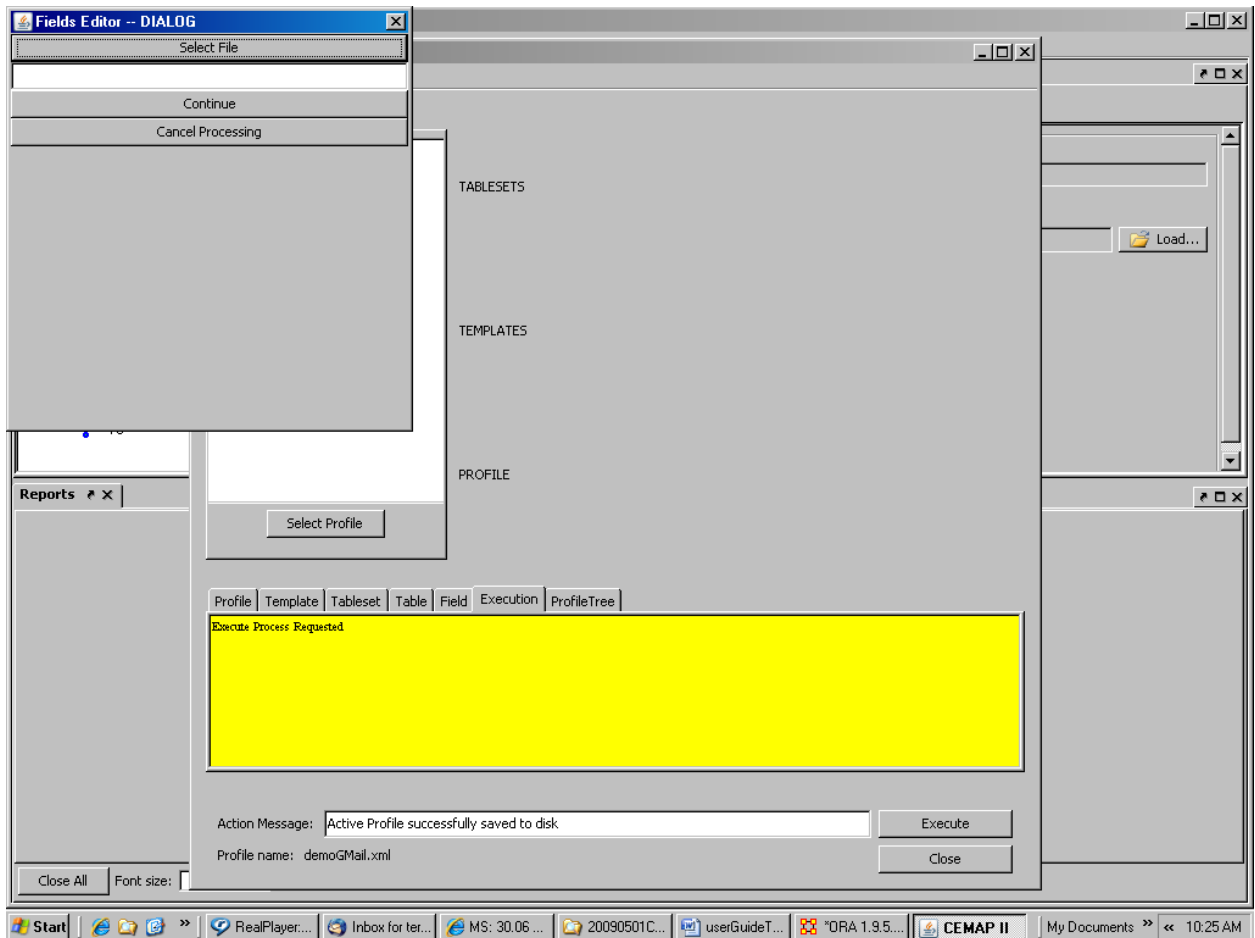


Figure 42. Result of Step 28.

Step 29:

Task: Bypass the opportunity to save the output network file to the disk

Action: Left click the Continue button

Result: The execute progress window will appear, e.g. Fig. 43

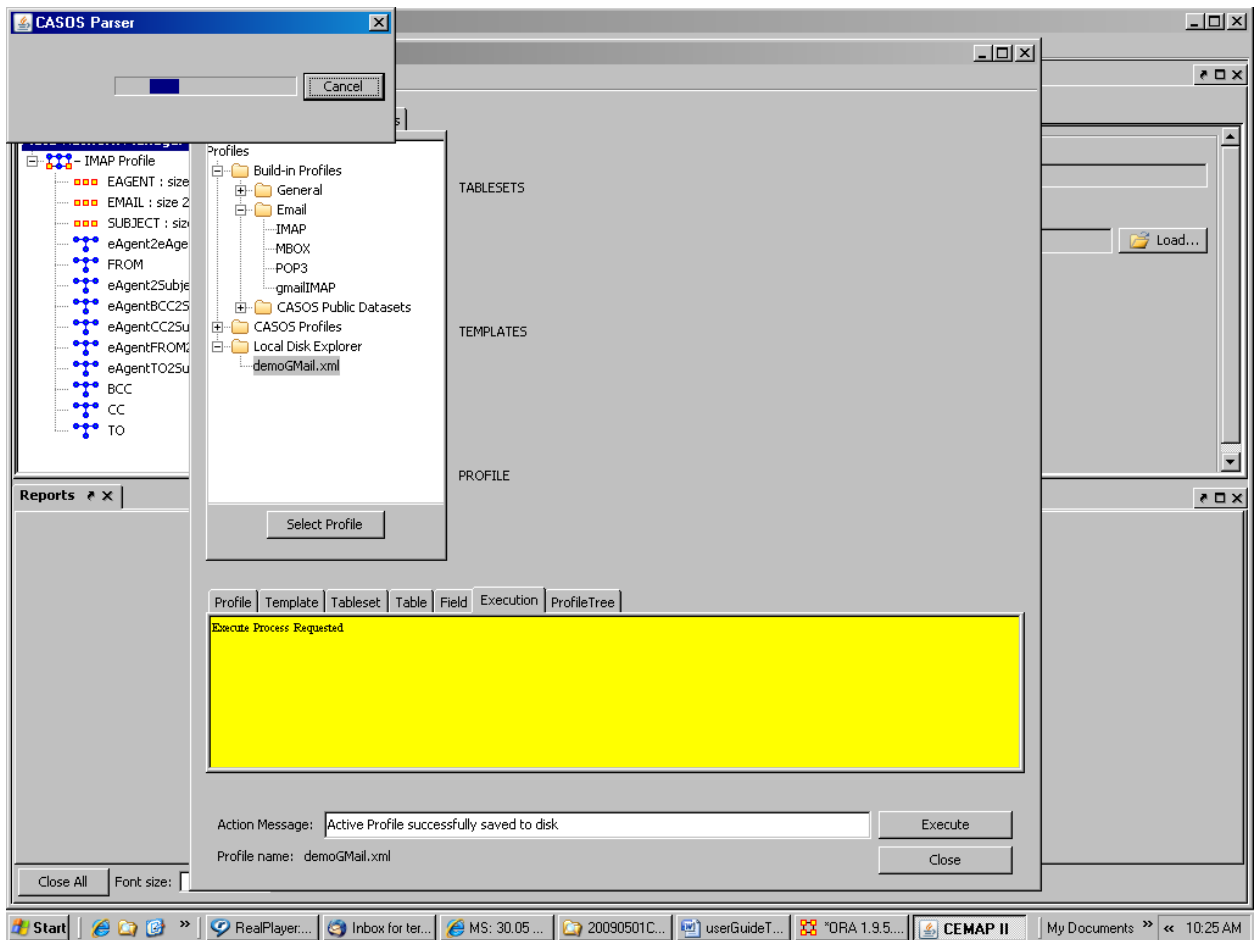


Figure 43. Result of Step 29.

Step 30:

Task: Profile is executing

Action: Wait until the progress bar disappears and the Parsing Complete message window appears

Result: The Parsing Complete message window will appear, e.g. Fig. 44

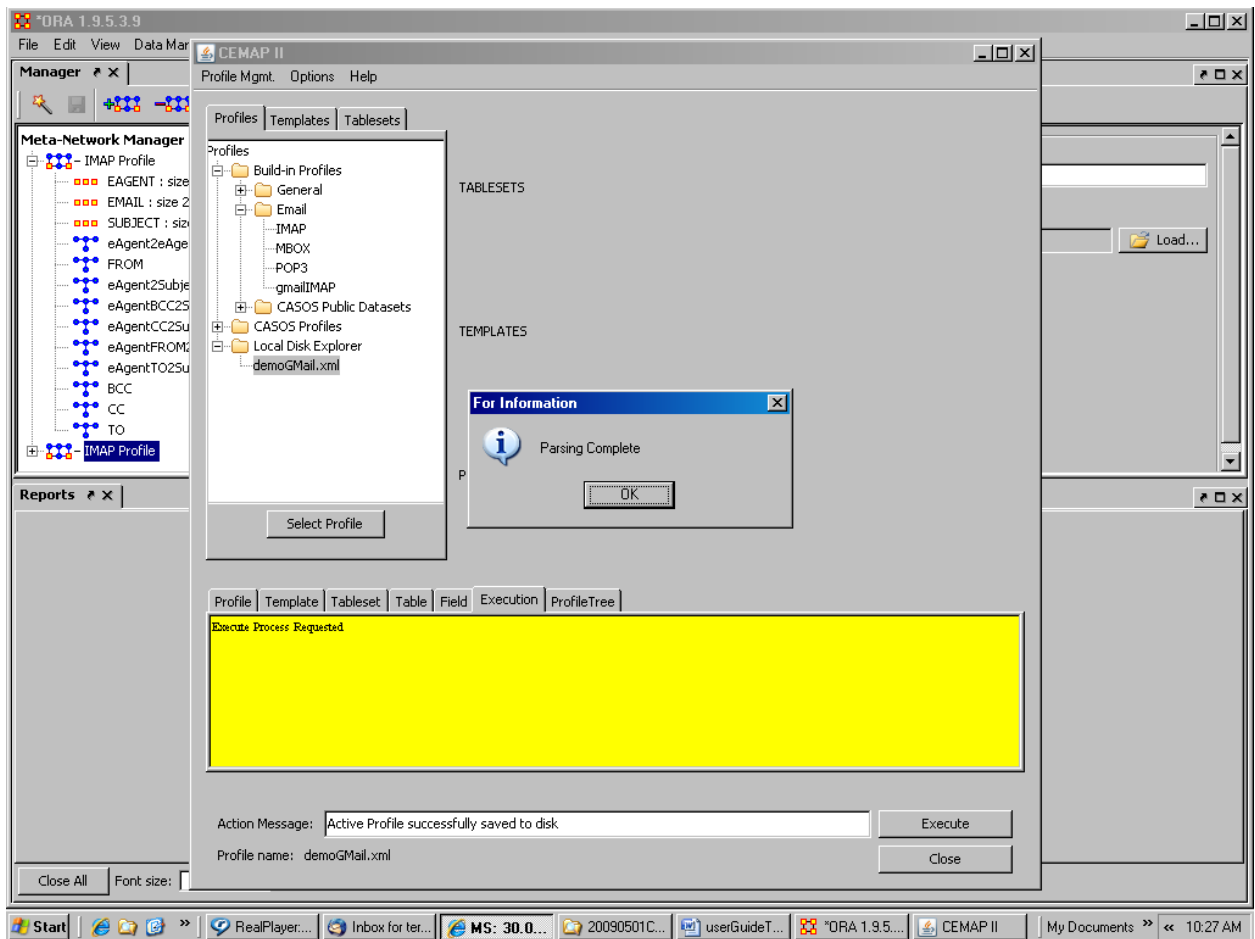


Figure 44. Result of Step 30.

Step 31:

Task: Acknowledge that parsing is complete

Action: Left click OK button in the Parsing Complete message window

Result: The Parsing Complete message window will disappear, e.g. Fig. 45

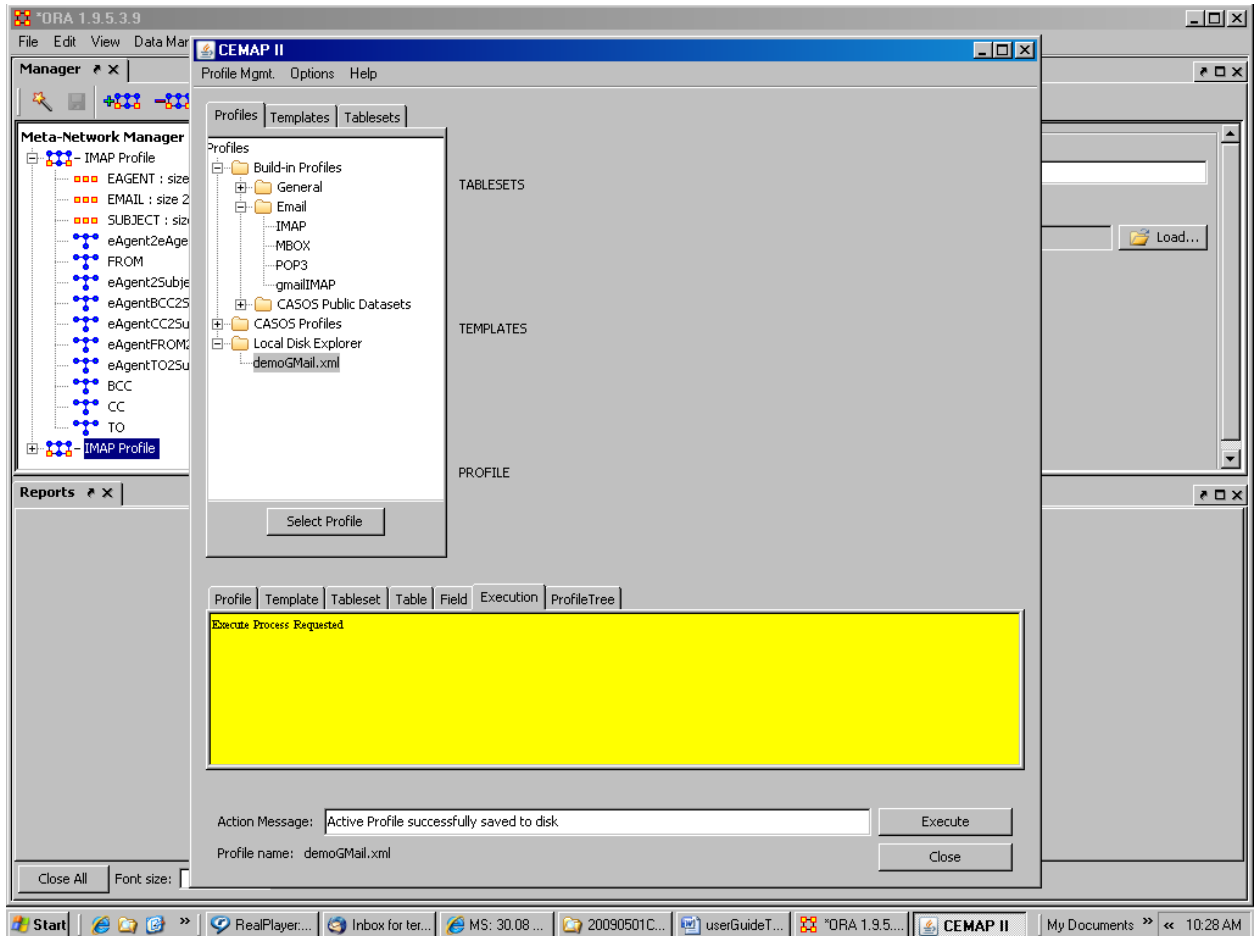


Figure 45. Result of Step 31.

Step 32:

Task: Go to ORA and explore!!!!!!

Action: Left click the second IMAP Profile network in the ORA files tree

Result: ORA will supersede the CEMAP window, e.g. Fig. 46

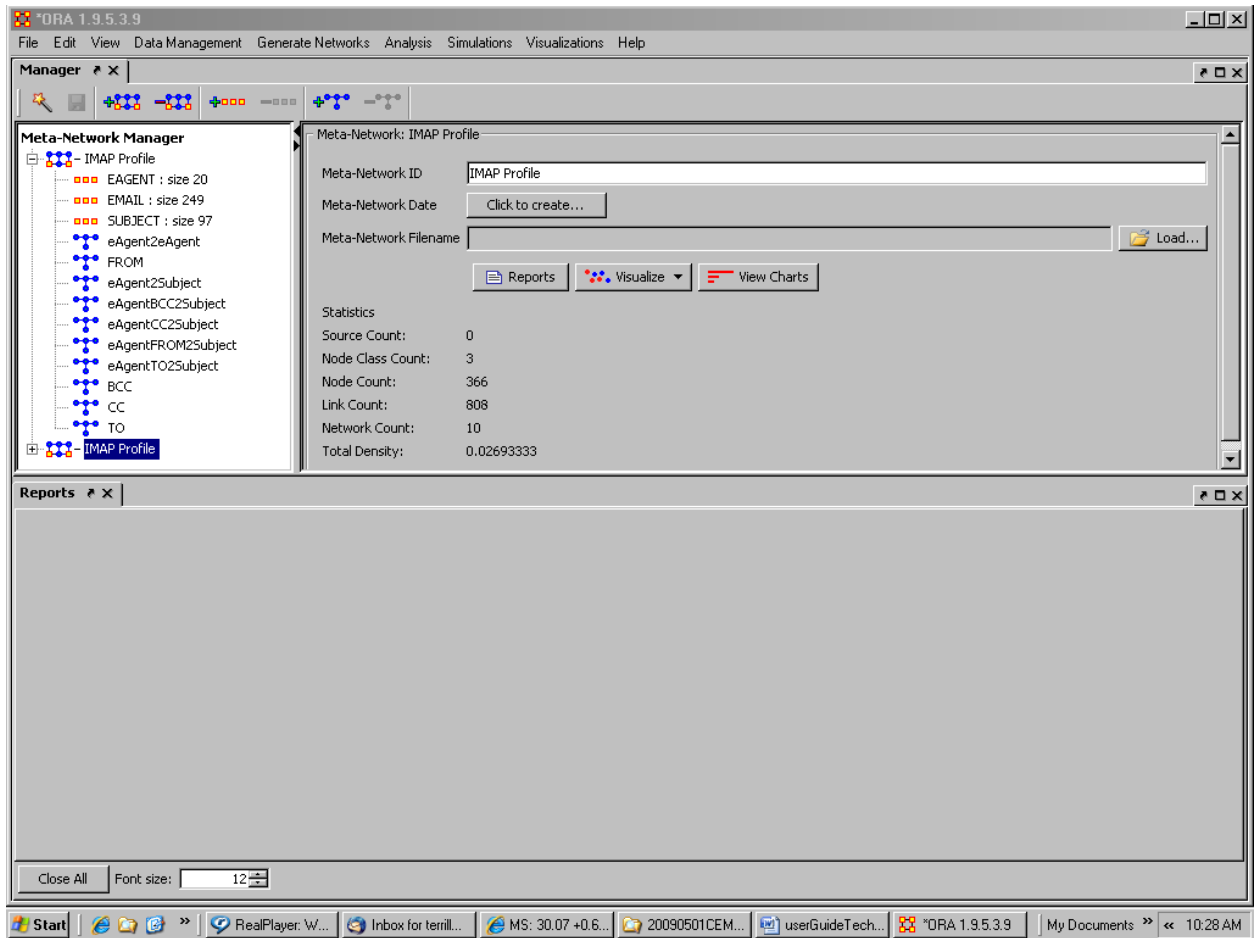


Figure 46. Result of Step 32.

3 Getting Started as a User: Managing Profiles

An operator in the role of a user is principally seeking to perform the actual CEMAP data loading process so that they can analyze the data. Generally, a user will want to merely load a profile and execute it. Although, they may also chose to modify the template according to their immediate needs. For example, perhaps the profile is set up to extract both the header and the text information from an email (see Frantz & Carley, 2008b) and the user wants to merely study the header network, and thus does not want the email texts to be extracted (typically this is input to AutoMap). So, in this case, the user would load the profile, modify it and then execute it. However, often times the user may want to save the modified profile and use the new profile in later sessions, by simply loading the modified profile and executing it.

3.1 Description of a Profile

A profile is a program file (in xml format) with the main purpose of keeping the completed mappings of a tableset and template(s). It can be thought of as the set of non-procedural instructions that CEMAP is to carry out when executing. There are usually the three logical aspects of a process fully defined within a profile. This is a CEMAP file that has a set of template(s), a set of tableset(s), and the mapping between the two. A mapping without a tableset is impossible. A mapping is always associated with a template. The profile can have any, all, or none of the fields for the template(s) or tableset(s) completed or not. A profile often corresponds with a routine task that the user performs often, periodically, or via a scheduled batch execution process. The profile is meant to “remember” all of the details pertinent to the CEMAP task, also some details can be purposely left unsatisfied by the creator of the profile (the actual name-location of the output file, perhaps, among other things like high-confidential passwords and such).

3.2 Managing Profiles

Profiles can be loaded, changed and executed. There are useful start-up profiles available via the built-in Profiles in CEMAP as well as other local or Internet-accessible locations. Moreover, users can create (usually by copying a start-up profile and modifying it) and store the change profile on their local computer disk.

3.2.1 Loading a profile

To load a profile, the top left window with the Profiles tab active will indicate the profiles available for loading. To load any profile from this window, highlight the desired profile and press Select Profile (see Fig 47.). The information in the profile (tablesets, templates and mappings) will be loaded into CEMAPs memory.

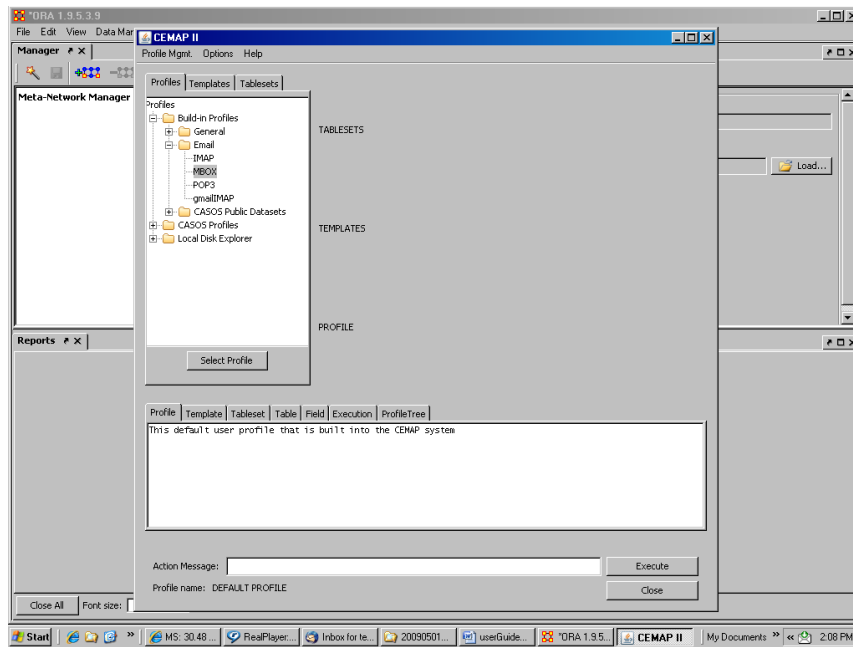


Figure 47. To load a profile into active memory: indicate profile item and press Select Profile.

3.2.2 Saving a profile

To save a profile that is currently in CEMAP's memory, go to the Profile Mgmt. menu (see Fig. 48 at the top of the CEMAP window). Left click the "Save active profile to files as..." and indicate where the profile should be saved. Any filename can be used, and it is suggested that the file extension be ".xml", though this is not necessary for CEMAP.

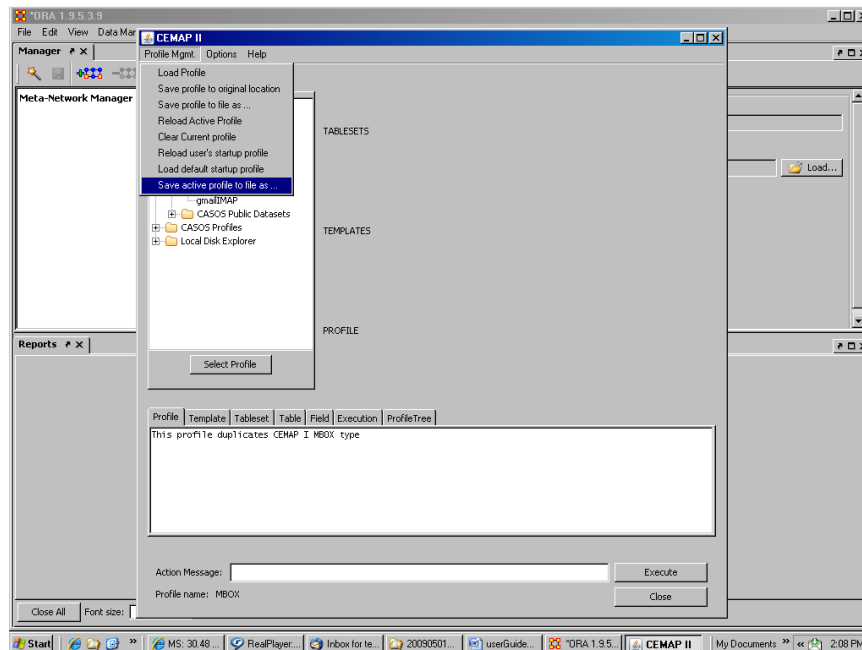


Figure 48. Save active profile by going to Profile Mgmt. menu.

3.2.3 Modifying a profile

To modify a profile that is in CEMAP memory, use the top left window to make changes to templates (see Fig. 49) and tablesets (see Fig. 50). To change a mapping, select the template item that is to be changed, by activating the template window table, and left clicking the item to be changed. Next, select the templates window tab and select the table that represents the data that should be applied to the template. By doing these two steps, the panel on the right side of the CEMAP window will display both the template item and the tableset columns. This area will also show the current mappings for that template item. To make the mapping change, drag the column header from the tableset portion to the column in the template where the data should be applied.

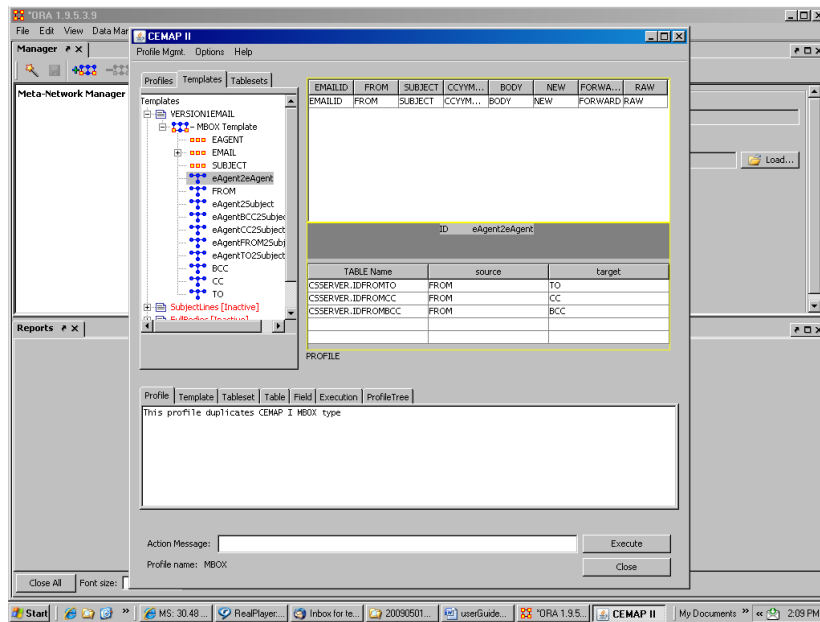


Figure 49. Modify a profile; in this case modifying a template.

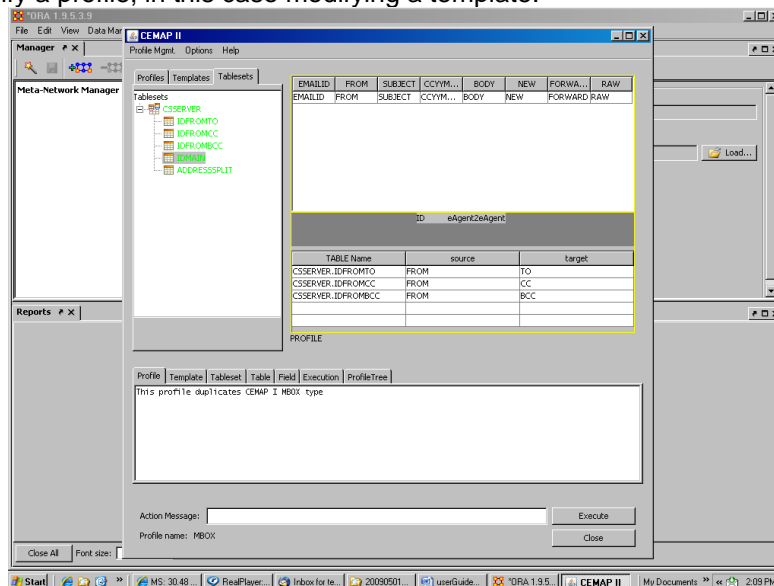


Figure 50. Modify a profile; in this case modifying a tableset.

The operator notes associated with a profile can also be changed. To change operator notes (those notes displayed on the bottom portion of the CEMAP window, simply type in your changes into the display window. Be sure to save any changes to the active profile, if appropriate.

3.3 Executing a Profile

For a profile to be executed, it must first be loaded, and thus in CEMAP's memory; when loaded into memory this profile is called an *active* profile. To execute an active profile, from the interactive GUI, left click the Execute button on the CEMAP window (see Fig. 51). Execute will begin immediately if all input fields are completed, if required by the template or tableset. A progress bar will be displayed showing the progress of the execute process. In the CEMAP batch command-line mode, a profile is automatically loaded into memory and executed, essentially in one step.

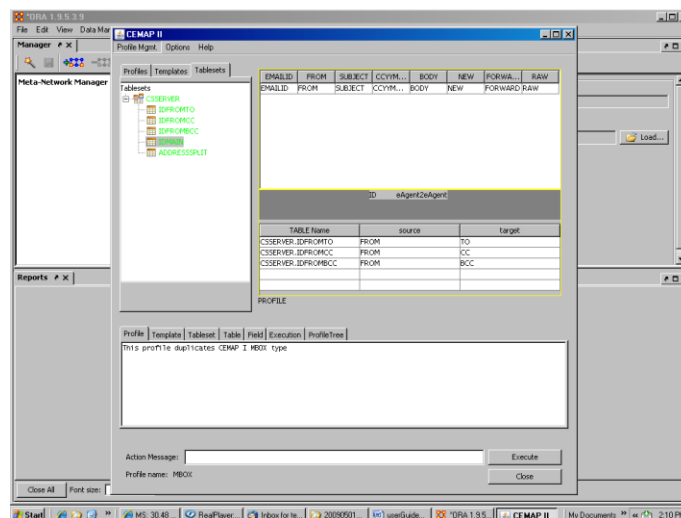


Figure 51. Execute a profile by left clicking the Execute button.

3.3.1 Input Fields

When an active profile is being executed, CEMAP will first check the input field requirements for the active templates and tablesets before reaching out to obtain the source data. Should CEMAP determine that any template or tableset fields necessitate a completed value and are currently incomplete, CEMAP will prompt the operator to complete the necessary fields before continuing on to execute the profile. Any of the fields entered by the operator during the execute process are loaded into active memory, so should the operator save the active profile to current field values will be saved as part of that profile. In batch mode, CEMAP will abruptly end without the operator being presented with alternatives. Figure 52 shows the menu item for entering field values for a template (Fig. 53) or a tableset (Fig. 54) should the operator desire to complete the field values before requesting the execute to take place; this is also done when operator is creating a profile for storing.

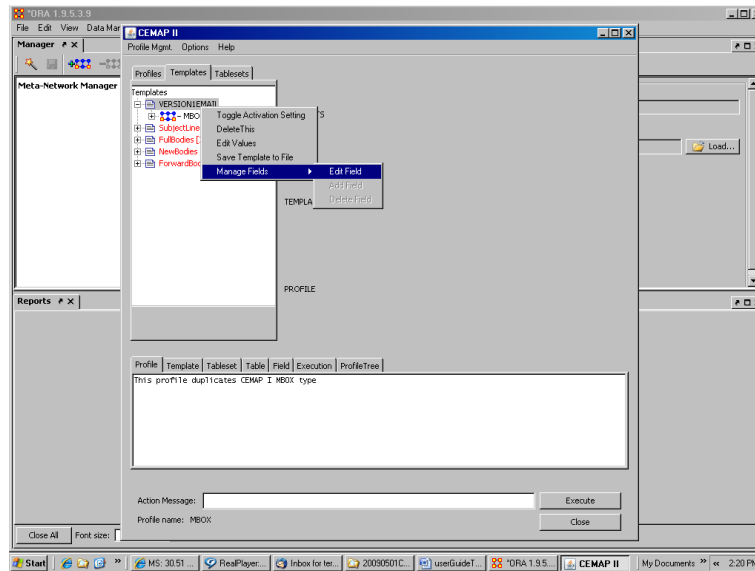


Figure 52. Complete the edit fields prior to execution, if desired.

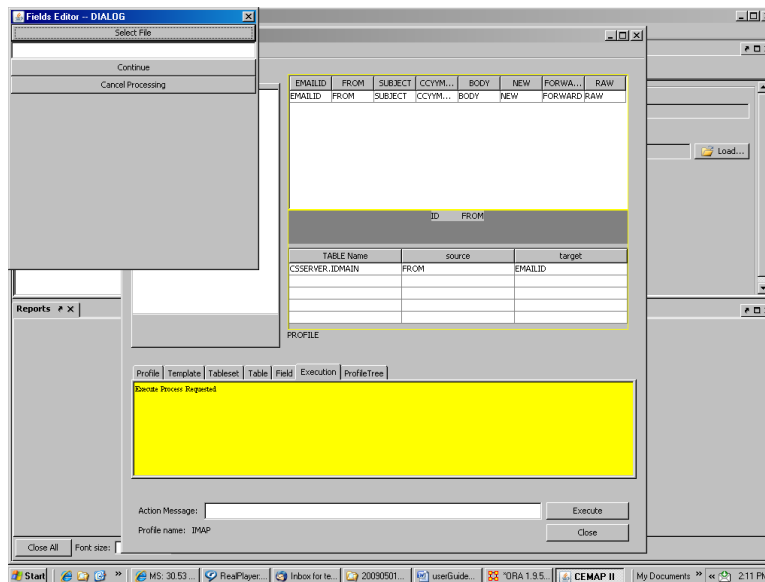


Figure 53. Template edit fields window is displayed and awaiting operator input.

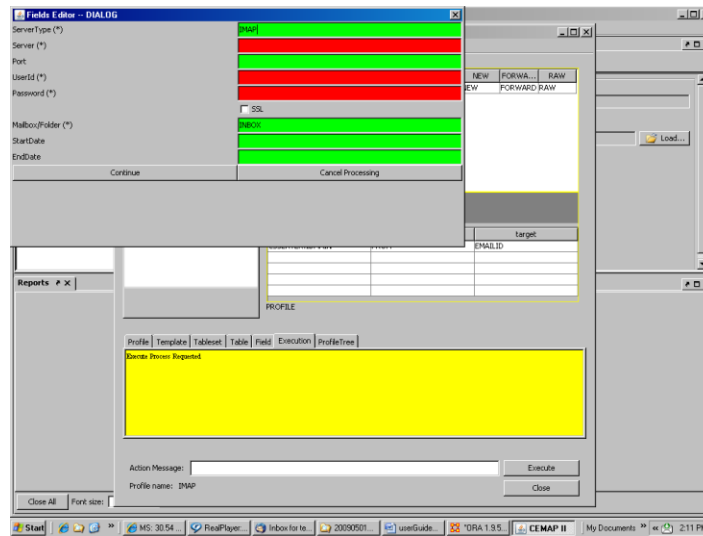


Figure 54. Tableset edit fields window is displayed and awaiting operator input.

3.4 The Start-up Profile

When a user starts their CEMAP session via a host, standalone, or batch, a special profile is loaded into CEMAP. For a first-time user, a default resource profile is loaded that comes installed with ORA or AutoMap. If the operator has previously saved a profile to a special location and name, it will be loaded into CEMAP upon start-up. This resource profile will have references to the default profile as provided by CEMAP. These profile are not user created profiles, but basic email and standard other profiles that the beginning user might be interested in using straight off. This resource file is not changeable by the CEMAP operator; however, the operator can create a custom resource file that does allow for customization of the CEMAP start-up. Whenever this operator starts CEMAP, it will be this customized resources profile that is loaded at the start. This user-custom start-up resources profile is stored on the user's local hard drive disk in the root user directory (C:\Documents and Settings\yourUserName). The name of the start-up file is ".CASOS_CEMAP_Profile.xml"

3.5 Mapping a Template to a Tableset

This data is made available to CEMAP via one or more tables defined collectively as a tableset. The process by which the output template is associated with the tableset table is call "mapping." The mapping process is simplistic with very few, but strict, rules. The operator specifies the specific mappings between the templates and the tablesets and has a flexible drag and drop mechanism in the CEMAP GUI to carry out this mapping process (see Fig. 55).

To set the template to tableset mappings for an active profile, the user will identify the template field that they are working on; this template fields is the output field in the DyNetML or unstructured text file. In the case of DyNetML file, this output field could be a node, a node attribute, a link, or a LOOM trailset, among other common possibilities. Next the operator indicates the specific table within a specific tableset that

will contain the data that should be mapped to the output field during the execution of the profile. Once the table is identified, a specific column from the table is chosen and it is this exact column, table, and tableset combination that is associated with the output field. Once all the mappings are made, a complete profile has been created. The profile can be stored for use later, or instead it can be executed immediately, with is most often the case.

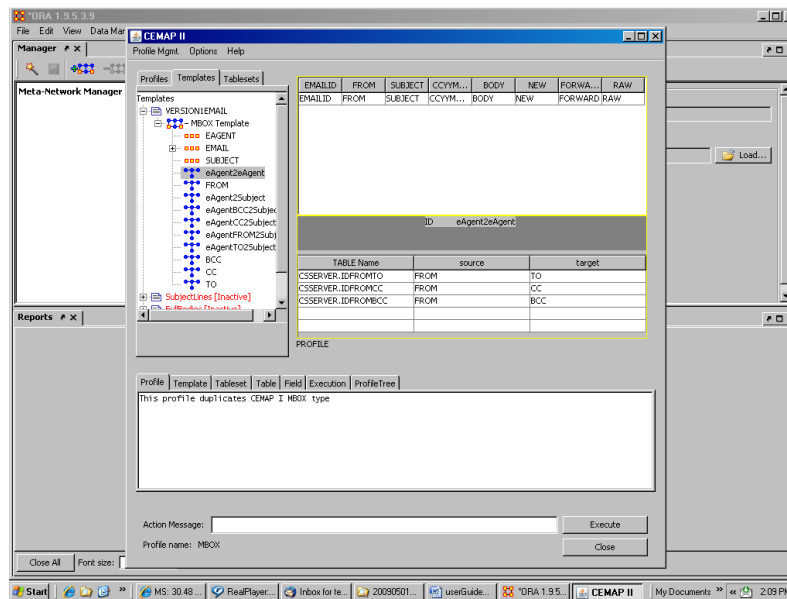


Figure 55. The top right quadrant to the CEMAP window is for mapping templates to tablesets.

3.6 Executing a Profile directly from the Command-Line

A profile can be executed in a batch processing mode by using the command-line process for running CEMAP. See the section above for specifics on how to execute CEMAP in command-line batch mode. Executing a profile from the command-line will not give the operator an opportunity to modify the profile before execution in CEMAP. Although, a stream editor, e.g., `sed`, can be utilized to change the profile in stdin or in a temporary file prior to offering it to CEMAP.

4 Getting Started as an Analyst: Managing Templates

Operators in the analyst role are those that focus on the design of templates, that is they are interested specifically in describing the form of the output from CEMAP that is consistent with what data they was to present to the downstream software, e.g. ORA and/or AutoMap. Analysts will spend most of their time and though process maintaining the CEMAP templates.

4.1 Description of a Template

A template is a description of an output format for CEMAP. These outputs most often consist of DyNetML templates, but there are numerous output templates that describe other output formats. Most of the output templates can be customized in some manner, according to the designer of the template. For example a DyNetML template allows the user to add, delete or change the characteristics, of a node class, or network, among other

several features of ample flexibility. The user has a great amount of flexibility in the flat file template as they are not constrained by the strict requirements of DyNetML. For example, a template can be customized to allow for the user to create a CSV file to import the output data into an Excel spreadsheet. This is merely by-product of the CEMAP feature set, which is designed expressly for the CASOS software suite.

4.2 Managing Templates

To manage a template the operator navigates in the top left window within the templates tab. This is where the operator can add, delete and maintain the template for the profile that is in CEMAP active memory. The operator uses the tree navigation buttons to navigate around the templates and within them. The left and right clicking of an item in this tree structure will present a menu of actions that are available to the operator, depending on the type of item being clicked (see Figs. 56 – 59). If the operator clicks to Templates item (root), new templates can be added to the profile. If the operator clicks a specific template item, then options relevant to that template will be presented. If a DyNetML template, options to add node classes and networks will be presented; if the item is a directory template, then the options for adding fields will be presented. In any template, their will always be a “Manage Fields” menu option will be available. This allows the operator to add, delete and set fields for the template.

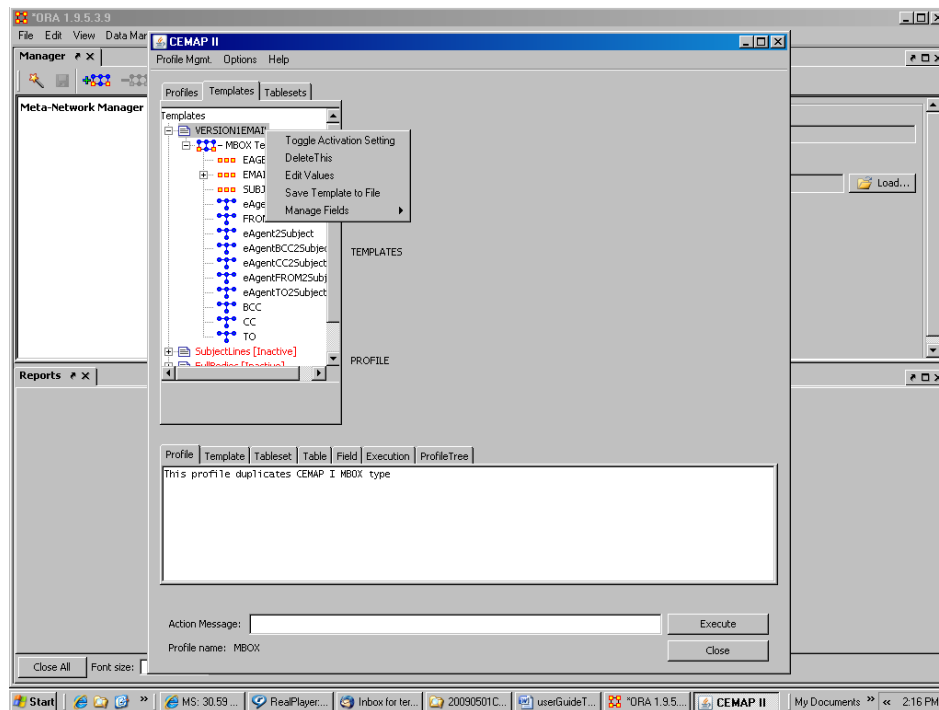


Figure 56. Template management menu.

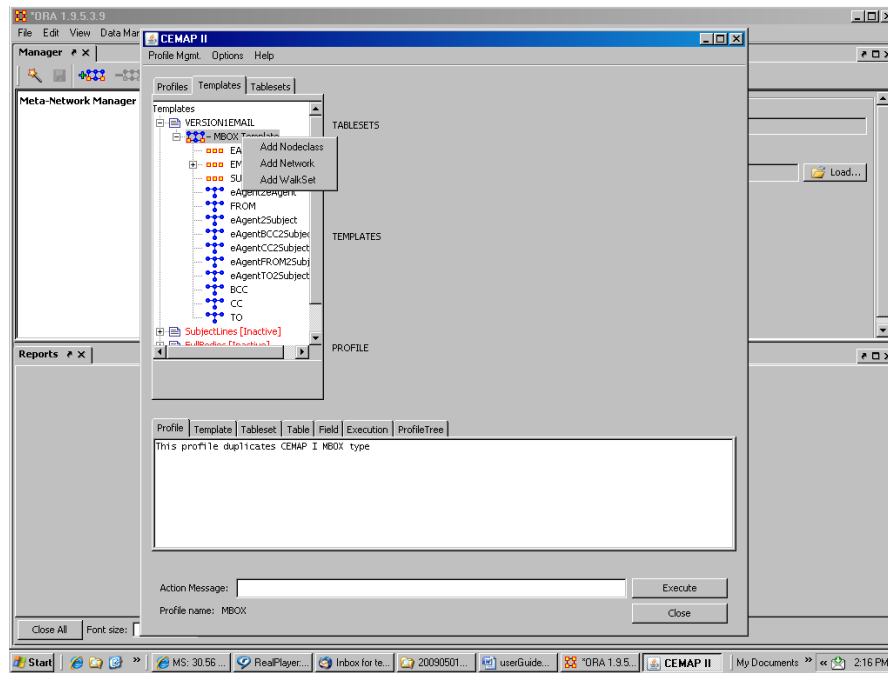


Figure 57. DyNetML template management menu.

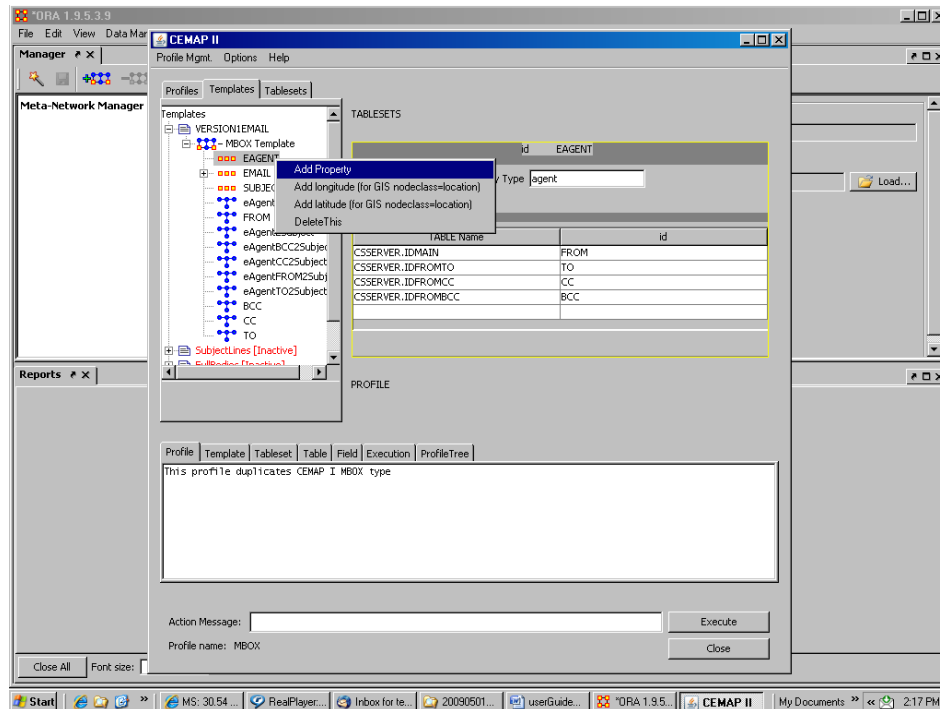


Figure 58. DyNetML node class management menu.

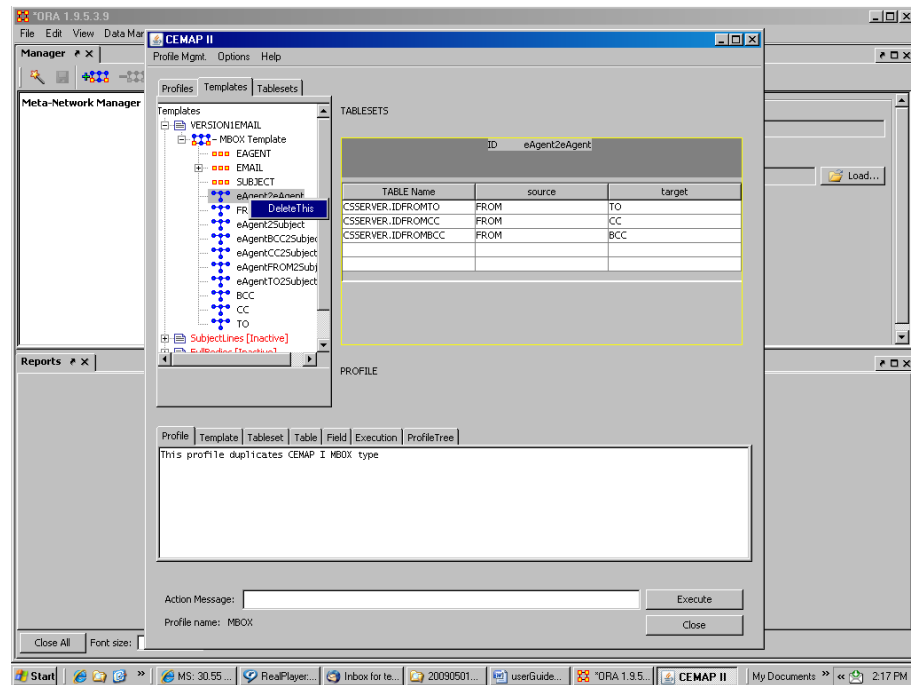


Figure 59. DyNetML template network management menu.

4.2.1 Template Fields

The fields feature in these documents is an xml element that is uniquely special in that it can be present at any level of the xml document structure. The field element can be, and often is at least at the top level of the document. (The component notes feature described in the prior sub-section, is in fact operationalized as a component field.) At the top level, the fields feature serves the users best to always (but not required) to have a Notes field. The field contains from zero to many field elements. The field elements are very specific to their host parent element and can be used to contain information such as execution parameters, file locations, etc. When at the top level of the document, a Notes fields is used to describe the contents of the document in terms that a user will understand and appreciate. The value of the Notes field will be presented to the users in the interactive usage of CEMAP when they are looking through the resourceLocator for a resource file.

The fields feature in these documents is an xml element that is uniquely special in that it can be present at any level of the xml document structure. The field element can be, and often is at least at the top level of the document. (The component notes feature described in the prior sub-section, is in fact operationalized as a component field.) At the top level, the fields feature serves the users best to always (but not required) to have a Notes field. The field contains from zero to many field elements. The field elements are very specific to their host parent element and can be used to contain information such as execution parameters, file locations, etc. When at the top level of the document, a Notes fields is used to describe the contents of the document in terms that a user will understand and appreciate.

Specific to templates, the fields are the necessary input parameters for creating the tableset output, through the underlying template. For example, creating output data this is

a network file, the operator will necessarily need to provide the name of the output file before CEMAP can execute the template. When a template is executed in CEMAP, the fields and their values will be reviewed and if all the necessary fields for the template are completed as required by the template, the operator will simply not be presented with the field entry window; alternatively the operator can complete the fields within having to execute the profile (see Fig. 60). According to the specifications set by the template, the fields window can be bypassed altogether when executing a template within a profile.

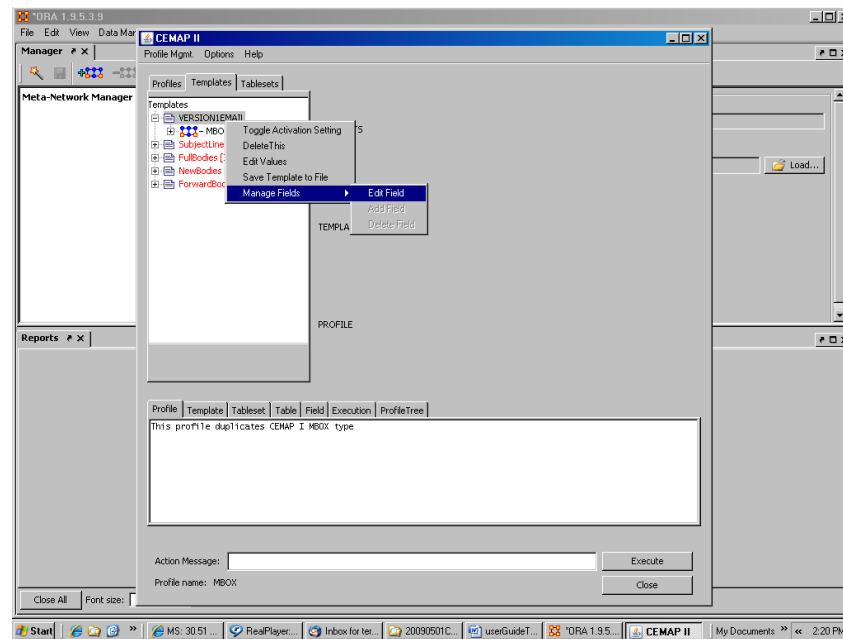


Figure 60. Completing template fields on demand.

4.3 Template Types

Currently there are three types of templates available to CEMAP operators. It is likely that this number will increase in the future as CEMAP continues its development. They are the primary methods for getting data into ORA and AutoMap software.

4.3.1 *DyNetML*

This template produces data in the format specific for use in ORA. The output that this template supports is an XML document that specifies a single meta-network. This template will always have a `networkFile` field that indicates where on the disk the network file should be placed. This can be blank to signify to not store the network on disk. In the special case of ORA, if CEMAP is being hosted by ORA, the network will always be placed into ORA memory, regardless of the value for the `networkFile` field.

4.3.2 *Directory*

This template produces data in a flat file format that is usually input to AutoMap. Multiple files are created and placed into the specified disk folder. The files can each be made up of a single data field, or have several fields delimited by a character of the operators choice – this is specified in the template's `delimiter` field.

4.3.3 Directory DyNetML

This template produces data in the format specific for use in ORA. The output that this template supports is an XML document that specifies a single meta-network, just as the DyNetML template produces, but this template creates multiple DyNetML files that are placed into a disk folder. Unlike the DyNetML template, the output will not automatically be loaded into ORA if ORA is the CEMAP host.

5 Getting Started as a Developer: Managing Tablesets

There are four primary component parts to the CEMAP architecture and user-design that even than the most basic user (a user that only is concerned with completed profiles) of CEMAP should be somewhat familiar with. The four components are described in this section and show in the Fig. 61 graphic below.

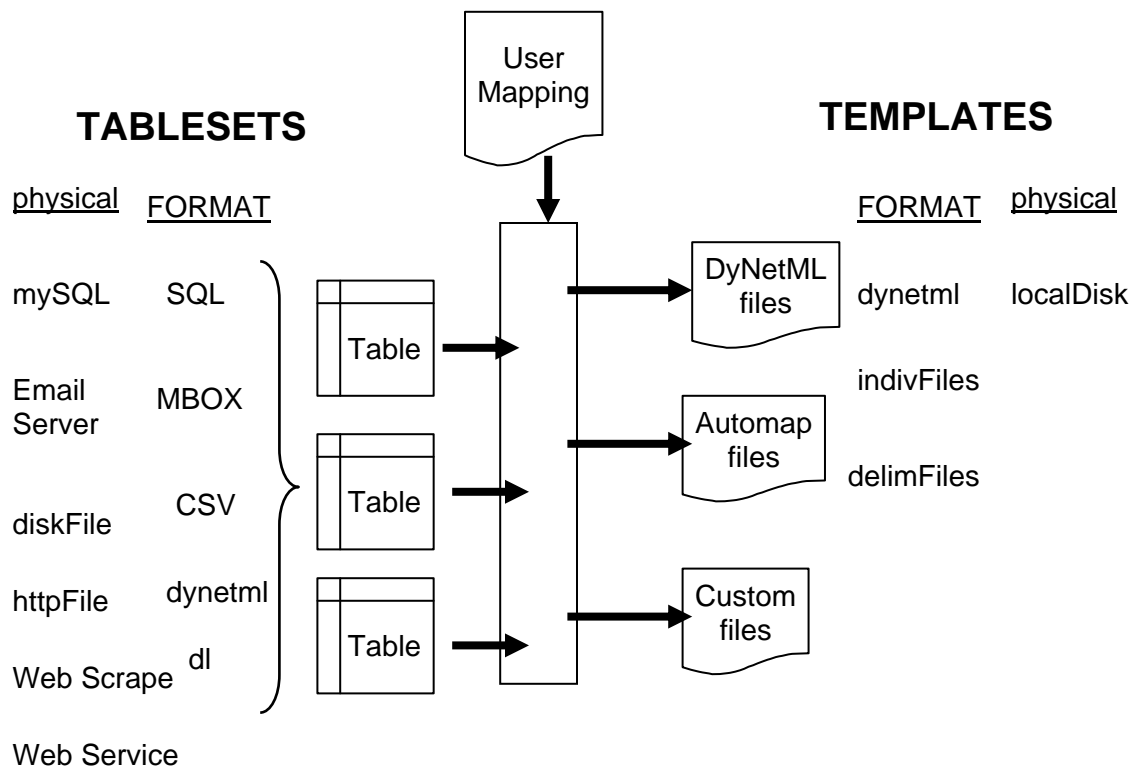


Figure 61. High-level architecture of CEMAP; All combined is a “Profile”.

5.1 Description of a Tableset

A tableset is a collection of tables that logically belong together either in the manner in which the original data is stored, or from the perspective of the user. For example, a collection of SQL statements involving a single database host and password, etc, can situation for a tableset, with each table corresponding to an SQL statement, but with the same characteristics as the other tables within the same tableset. A tableset translates the physical data source into a row-column oriented table that can be mapped with one or more other tables or template. A tableset takes much., most, or all of the complexity out

of a user reaching an original data source for their subsequent ORA/AutoMap analysis (for ORA/AutoMap interoperability, see Carley, Diesner, Reminga, & Tsvetovat, 2004).

The tableset construct is a form that situates CEMAP for an unending amount of flexibility in interfacing with real-world data. The underlying code for each tableset deals with all of the peculiarities of the real-world data and transforms that data into a table, or set of tables. This is all the tableset designer, or developer, needs to be concerned with. There is no notion of a network or DyNetML to the tableset view.

5.2 Managing Tablesets

A tableset is a collection of tables that logically belong together either in the manner in which the original data is stored, or from the perspective of the user. For example, a collection of SQL statements involving a single database host and password, etc, can situation for a tableset, with each table corresponding to an SQL statement, but with the same characteristics as the other tables within the same tableset. A tableset translates the physical data source into a row-column oriented table that can be mapped with one or more other tables or template. A tableset takes much, most, or all of the complexity out of a user reaching an original data source for their subsequent ORA/AutoMap analysis.

The tableset construct is a form that situates CEMAP for an unending amount of flexibility in interfacing with real-world data. Operators can add manage tablesets in the GUI (see Figs. 62-64). The underlying code for each tableset deals with all of the peculiarities of the real-world data and transforms that data into a table, or set of tables. This is all the tableset designer, or developer, needs to be concerned with. There is no notion of a network or DyNetML to the tableset view.

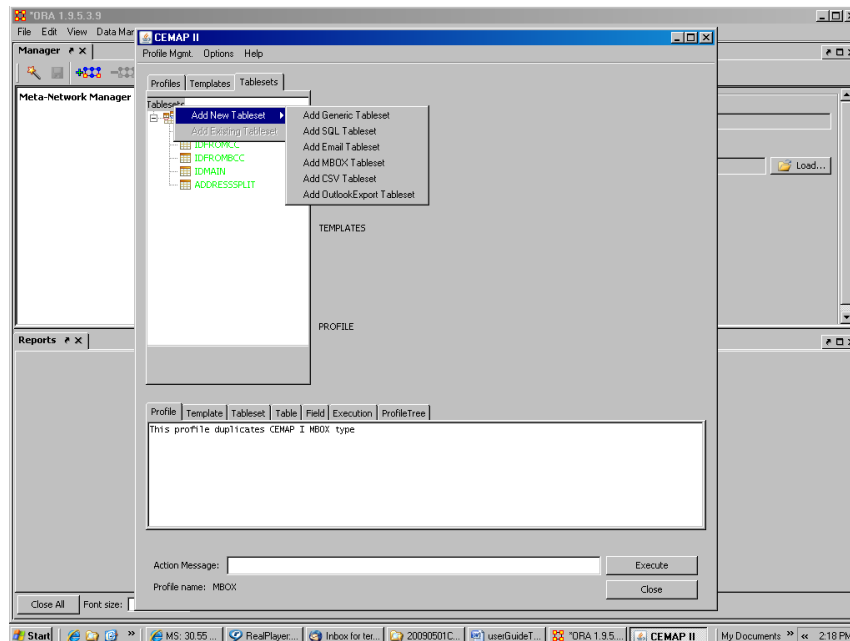


Figure 62. Managing tablesets.

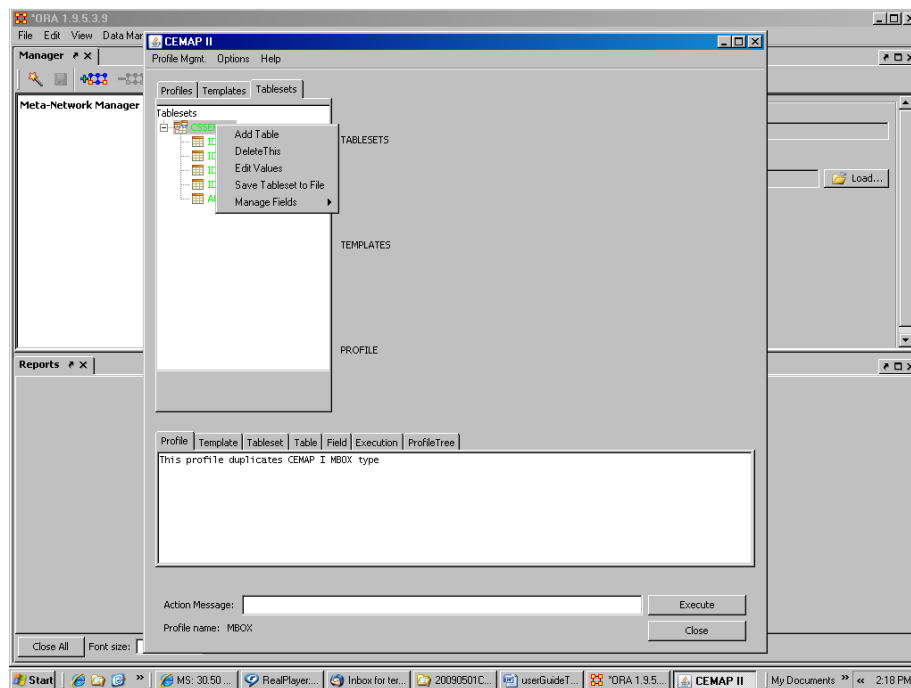


Figure 63. Managing a tablesset.

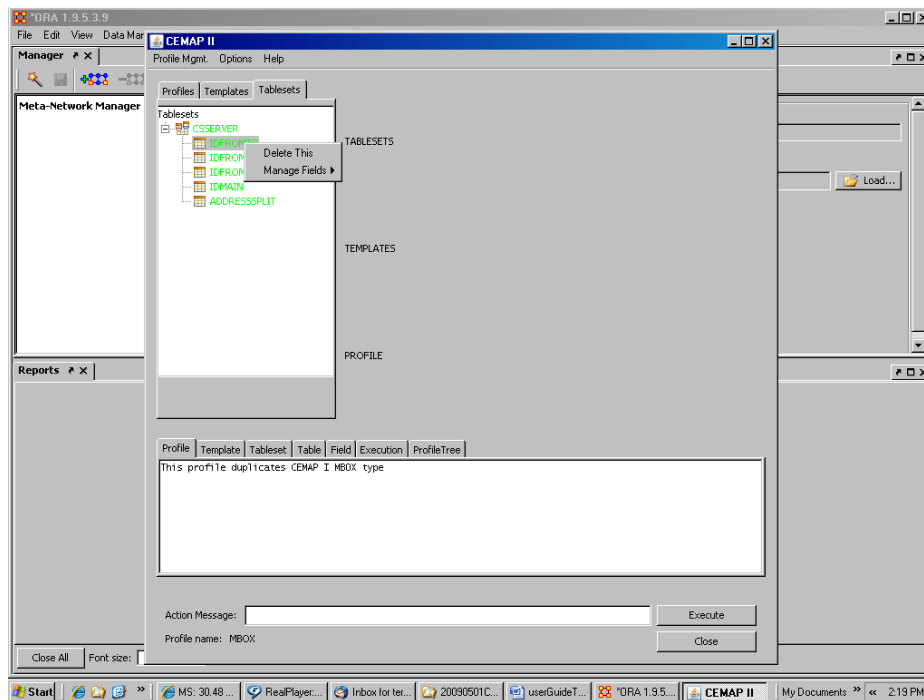


Figure 64. Managing a table.

5.2.1 Tablesset Fields

The fields feature in these documents is an xml element that is uniquely special in that it can be present at any level of the xml document structure. The field element can be, and

often is at least at the top level of the document. (The component notes feature described in the prior sub-section, is in fact operationalized as a component field.) At the top level, the fields feature serves the users best to always (but not required) to have a Notes field. The field contains from zero to many field elements. The field elements are very specific to their host parent element and can be used to contain information such as execution parameters, file locations, etc. When at the top level of the document, a Notes fields is used to describe the contents of the document in terms that a user will understand and appreciate.

Specific to tablesets, the fields are the necessary input parameters for creating the tableset, through the underlying tables and source data location and formats. For example, obtaining data from an email server requires at least identifying the email server, the type of server (IMAP, POP, etc), a user id, and a password; sometimes, more information is required as per the server. Moreover, these fields that are exposed to the operator, or user of the tableset, can be set to have default values so that the operator need only to review the fields (see Figs. 65-66). When a tableset is executed in CEMAP, the fields and their values will be reviewed and if all the necessary fields for the tableset are completed as required by the tableset, the operator will simply not be presented with the field entry window. According to the specifications set by the tableset, the fields window can be bypassed altogether when executing a tableset within a profile.

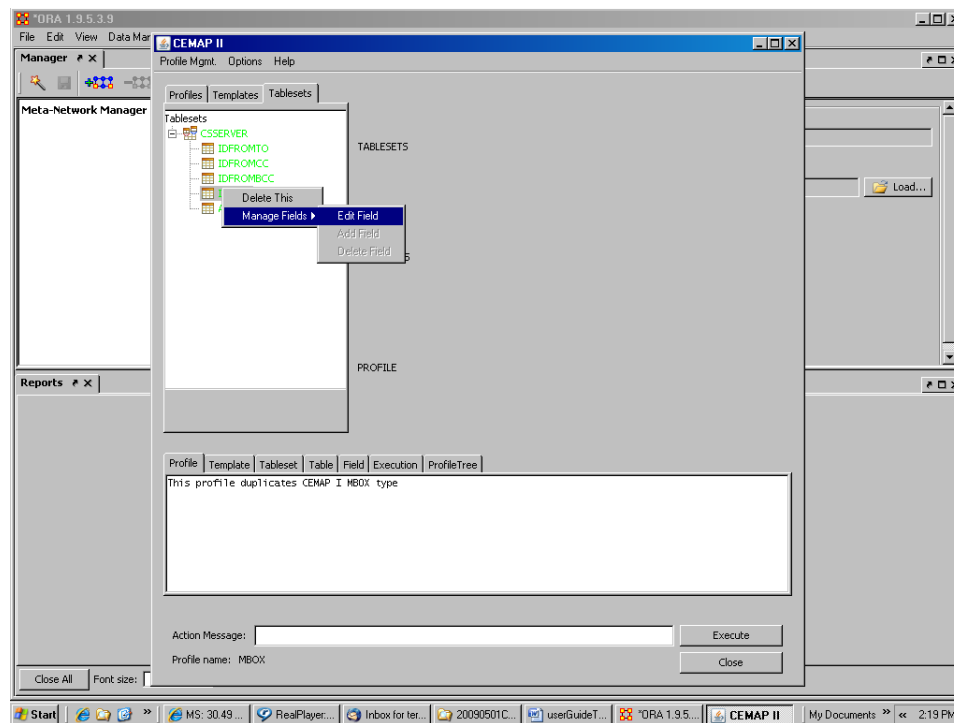


Figure 65. Changing a value for a tableset field.

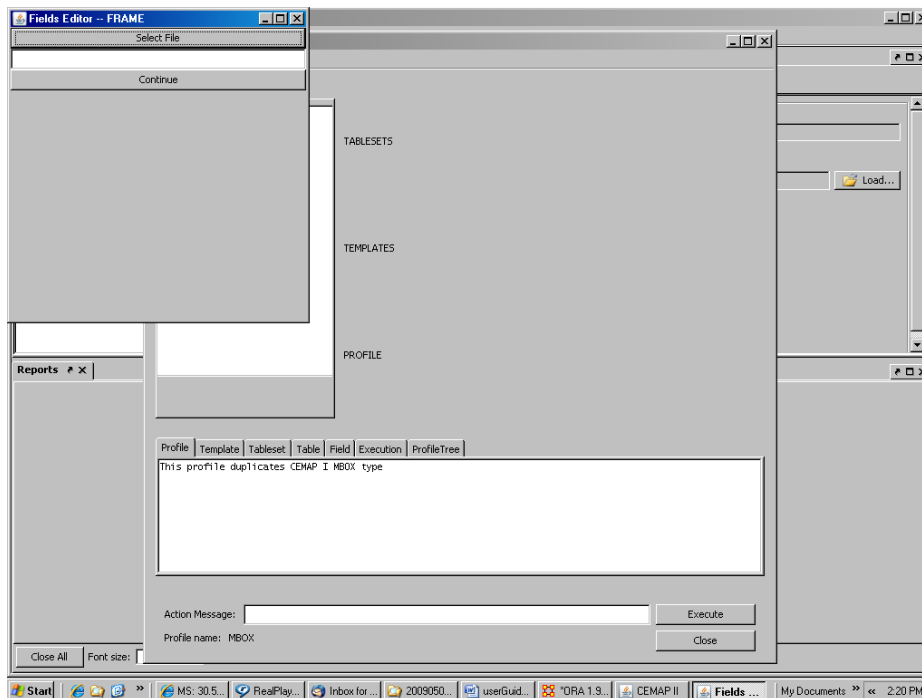


Figure 66. Completing a value for a tableset field.

5.3 Using Resources

A resource is a mapping of a physical file, data or CEMAP specific file (profile, tableset, template, or other resource file), that the user has access to. This resource can reside anywhere reachable to the user's computer, e.g. local disks, the Internet, etc. The function of a resource is to simplify the reaching of the file to the user. That is it saves the user from remembering long URL's or file paths, etc. There are two main types of resources, a resourceDirectory and a resourceFile. Locations for either can be on a local disk, or on the Internet. The resourceDirectory indicates the location of a group of resourceFiles, therefore can be a local disk directory, a local or Internet-based java jar file, a local or Internet-based zip file, or the CEMAP system files embedded within the CEMAP software. A resourceFile can be a file on the local disk, the Internet or within the java jar or zip file specified by a resourceDirectory. A resourceFile is an xml document file that has "xml" as its file name extension and contains the root xml element, <CasosCemap>.

6 Forward

CEMAP continues its development as this document is being written. Operator usability is paramount to the success of CEMAP and is a key design point of the architecture and graphical user interface. Moreover, the development of new templates will position CEMAP to being even more commonly used within the analyst community as more innovative tools are developed that build on the interoperability concept embraced by ORA and AutoMap. Tablesets are being developed and added to the inventory regularly and as each is added, the power of CEMAP increases as well. The benefits of using CEMAP will expand as the user base, and thus the accumulated development effort, grows.

7 References

- Carley, Kathleen, Diesner, Jana, Reminga, Jeffrey, Tsvetovat, Max. (2004). Interoperability of Dynamic Network Analysis Software.
- Carley, Kathleen & Reminga, Jeffrey. (2004). ORA: Organization Risk Analyzer. Carnegie Mellon University, School of Computer Science, Institute for Software Research International, Technical Report CMU-ISRI-04-106,
- Diesner, Jana & Carley, Kathleen. (2004). AutoMap1.2 - Extract, analyze, represent, and compare mental models from texts. Carnegie Mellon University, School of Computer Science, Institute for Software Research International, Technical Report CMU-ISRI-04-100
- Frantz, Terrill & Carley, Kathleen. (2008a). *CEMAP: An Architecture and Specifications to Facilitate the Importing of Real-World Data into the CASOS Software Suite*. Carnegie Mellon University, School of Computer Science, Institute for Software Research, Technical Report CMU-ISR-08-130
- Frantz, Terrill L. & Carley, Kathleen M. (2008b). Transforming raw-email data into social-network information, In Christopher C. Yang, Hsinchun Chen, Michael Chau, Kuiyu Chang, Sheau-Dong Lang, Patrick S. Chen, Raymond Hsieh, Daniel Zeng, Fei-Yue Wang, Kathleen Carley, Wenji Mao, and Justin Zhan (Eds.) (2008). 'Intelligence and Security Informatics Workshops, PAISI, PACCF and SOCO 2008' Springer, Lecture Notes in Computer Science, No. 5075. Pacific Asia Workshop on Intelligence and Security Informatics (PAISI 2008). Grand Formosa Regent Hotel , Taipei, Taiwan, 17 June 2008.
- Frantz, Terrill & Carley, Kathleen. (2008c). *Harvesting Ego-Network Data from Facebook*. Carnegie Mellon University, School of Computer Science, Institute for Software Research, Technical Report CMU-ISR-09-102
- Tsvetovat, Max, Reminga, Jeffrey & Carley, Kathleen. (2003). DyNetML: Interchange Format for Rich Social Network Data. NAACSOS Conference 2003, Day 2, Electronic Publication, Pittsburgh, PA.